AD-A091 600 GEORGE WASHINGTON UNIV WASHINGTON DC DEPT OF ELECTRI--ETC F/G 9/2
ANALYSIS AND DEVELOPMENT OF IMAGE STATISTICS AND REDUNDANCY REM--ETC(U)
SEP 80 M H LOEW, R L PICKHOLTZ, L GOLDMAN DAAK70-79-C-0147
UNCLASSIFIED GWU-EE-CS-80-09 ETL-0239 NL

1 of 2
AD
A091600

THE
GEORGE
WASHINGTON
UNIVERSITY

STUDENTS FACULTY STUDY R
ESEARCH DEVELOPMENT FUT
URE CAREER CREATIVITY CC
MMUNITY LEADERSHIP TECH
NOLOGY FRONTI ESIGN
ENGINEERING APP ENC
GEORGE WASHIN NIV

SCHOOL OF ENGINEERING
AND APPLIED SCIENCE

ETL-0239

ANALYSIS AND DEVELOPMENT OF IMAGE STATISTICS AND REDUNDANCY REMOVAL

Murray H. Loew
Raymond L. Pickholtz
Lee Goldman
Fred Hill
Fred Lawler
Joseph Van Meter

Department of Electrical Engineering and
Computer Science
The George Washington University
Washington, D. C. 20052

September 1980

Final Technical Report

Approved for public release; distribution unlimited

Prepared for

U. S. Army Engineer Topographic Laboratories
Fort Belvoir, VA 22060

⑨ Final rept. 15 Sep 79 - 14 Sep 80

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(19) REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER (18) ETL-0239	2. GOVT ACCESSION NO. AD-A091	3. RECIPIENT'S CATALOG NUMBER 600
4. TITLE (and Subtitle) (6) Analysis and Development of Image Statistics and Redundancy Removal		5. TYPE OF REPORT & PERIOD COVERED Final Technical Report Sept. 15, 1979-Sept. 14, 1980
		6. PERFORMING ORG. REPORT NUMBER (14) GWU-EE-CS-80-09
7. AUTHOR(s) (10) Murray H. Loew, Raymond L. Pickholtz, Lee Goldman, Fred Hill, Fred Lawler, Joseph Van Meter		8. CONTRACT OR GRANT NUMBER(s) (15) DAAK70-79-C-0147 NEW
9. PERFORMING ORGANIZATION NAME AND ADDRESS Dept. of Electrical Engineering & Computer Science George Washington University Washington, D.C. 20052		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army Engineer Topographic Laboratories Ft. Belvoir, VA 22060		12. REPORT DATE (11) Sep 1980
		13. NUMBER OF PAGES 139 plus appendices
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) (12) 62		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Pattern recognition; cartography; image processing; image coding; Hough transform; medial axis transform

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The goal of classifying objects of cartographic interest in aerial photographs was approached using techniques from pattern recognition and image processing. Bridge and airport images were chosen as the initial objects of interest and segments of photographs containing them were digitized for the data base. Edge-detection and Hough transform algorithms identified structures as candidate bridges; additional decision logic (using global contrast and other attributes) further reduced the set. Results indicate

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

411264

the feasibility and low computational cost of the approach.

Additional results in discrete medial-axis transformation are presented, as are methods for encoding the two kinds of images. The characteristics of the two kinds of targets are so distinctive that encoding promises substantial efficiencies.

PREFACE

The work described here was supported by the U.S. Army Engineer Topographic Laboratories under Contract DAAK70-79-C-0147 during the period September 15, 1979 - September 14, 1980.

The authors are grateful to Drs. Pi-Fuay Chen and Fred Rohde of ETL for their guidance and interest.

TABLE OF CONTENTS

ERRATA

Page 26, line 11: *For* B *read* $\underline{B}$
Page 29, line 12: *For* follows: *read* follows (where $\bar{\underline{B}}$ is the average brightness of the nine elements in $\underline{B}$):
Page 29, line 13: *For* $\sqrt{(\bar{\underline{B}} - B)\cdot(\bar{\underline{B}}\ \underline{B})}$ *read* $\sqrt{(\underline{B}-\bar{\underline{B}})\cdot(\underline{B}-\bar{\underline{B}})}$
Page 29, line 16: *For* $(\underline{B}-\underline{B})^2$ *read* $(\underline{B}-\bar{\underline{B}})^2$
Page 35, line 2: *For* space). *read* space.
Page 56, line 19: *For* $\underline{P}'$ *read* P'
Page 56, line 21: *For* $\bar{a}(P)$ *read* a(P')
Page 57, line 1: *For* P_i *read* P_i'
Page 57, line 2: *For* p'_i *read* P_i'
Page 57, line 5: *For* where Dq is *read* where D_q is
Page 58, line 14: *For* $[P_i]$ *read* $[P_i']$
Page 58, line 15: *For* P_i *read* P_i'
Page 59, line 11: *For* P_i *read* P_i'
Page 59, line 12: *For* P^1_1 *read* P'_1
Page 59, line 13: *For* P_i *read* P_i'
Page 59, line 20: *For* P_k *read* P_k'
Page 59, line 22: *For* P_k *read* P_k'
Page 69, line 22: *For* $0\bar{<}R\bar{<}R$ *read* $0<R'\bar{<}R$
Page 70, line 2: *For* $\bar{E}$ m-R' *read* E $\bar{<}$m-R'
Page 70, line 5: *For* p $\leq$ m-R' *read* p $\geq$ m-R'
Page 72, line 4 of caption: *For* (2)<E Test *read* and test that compares two neighbors only to E
Page 72, line 5 of caption: *For* 2<E-R' Test *read* using test that compares two neighbors to (E-R')
Page 79, line 2 of part d): *For*)0 *read* (0
Page 79, line 5 of part d): *For* LIne *read* Line
Page 101, line 21: *For* encoded. *read* encoded; the "X" column of the table corresponds to the leftmost column of Fig. 5.9 while the remainder of the entries in each row of the table correspond to the y-coordinates (top row of Fig. 5.9).
Page 104, line 6: *For* encoder. *read* encoder; the number preceding (above) the M indicates the chain-code direction and the number following (below) the M indicates the number of consecutive occurrences of a component having that direction.
Page 134, line 1: *For* completion *read* complete
Page 134, line 5: *For* largearea *read* large area

LIST OF FIGURES

LIST OF TABLES

ANALYSIS AND DEVELOPMENT OF IMAGE STATISTICS AND REDUNDANCY REMOVAL

CHAPTER I

INTRODUCTION AND BACKGROUND

Extracting features from aerial photographs has been an important cartographic effort for many decades. Much technology has been introduced to improve and refine this process, but the crucial and limiting factor has been the need for trained human operators with the intelligence, experience and skills to recognize and identify the many diverse (and sometimes unexpected) cartographically interesting objects that appear in a typical aerial photograph.

It would appear that many of the functions performed by a trained cartographic observer could and should be automated. At the very least, a computer aided human system would increase productivity and decrease the inevitable fatigue and thereby improve both the detection and false alarm rate of routine picture examination.

The process of extracting important features from a photograph at first glance, appears to be characterized within the broad framework of the theory of pattern recognition. Work on pattern recognition has been ongoing for about three decades with some notable successes. These successes have been evident in printed character recognition, biomedical applications such as cell anomaly identification, and to an

extent, in remote sensing. The application of pattern recognition theories to cartographic feature extraction is, however, very sparse indeed.

Much of the work on pattern recognition has been based on either

(1) statistical decision theoretic methods

(2) template matching (matched filters)

(3) syntactical decriptions of geometric objects

The statistical approach is based on knowledge of a priori probability characterization of the objects to be examined. This implies good statistical behavior of the ensemble of pictures to be examined and the objects to be characterized.

Template matching works quite well if again, there is a priori information about precise shape, size and orientation of objects to be identified.

Syntactical grammars constitute a relatively new, but more abstract way of dealing with the problem by translating the image into strings of predefined primitives and then performing the recognition operation on the picture description language so developed. Various approaches to the translation process has been proposed including linear strings, tree representations, webs and other data structures. These techniques have had some measure of success in pictures which can be skeletonized to line drawings or other forms of a similar nature. Notable are the successes in chromosome analysis.

It should be emphasized that all image analyses usually require a considerable amount of preprocessing such as linear transformations and enhancement techniques. Indeed, the vast majority of research on images has been done in this area of digital image processing which by itself cannot and does not yield either feature extraction or recognition of objects.

For aerial cartographic analysis, which is the subject of this report, it is unfortunate that at this point in its development, very little of the classical pattern recognition theories are immediately applicable. If we are to be successful with developing algorithms which actually work on real images, a good deal of ad hoc methods must be used. The reason for this is that it is rare that cartographically interesting objects can be assigned a priori statistical distributions such as is needed in a decision theoretic approach. Templates are not likely to be successful because even approximate shape for the same kind of object is too variable, let alone orientation variability. We are thus left to examining the basic nature of the objects of interest. In aerial catography, the basic nature can be dichotomized:

(1) Natural and gross man-made objects such as forests, fields, water, city streets, etc. These objects are properly characterized by the texture, reflectivity and fine structure.

(2) Discrete man-made objects such as bridges, roads, canals, airports, storage containers, specific industrial sites. These man-made objects have the distinguishing feature of having unique geometries, shape boundaries and fairly sharp contrast with their environment.

The natural and gross man-made objects probably will yield some degree of computerized recognition by taking advantage of the texture and fine structure. The techniques for doing so are available and indeed, Dr. P. F. Chen and his colleagues at the Engineer Topographic Laboratories have demonstrated a preliminary algorithm which discriminates forest, field, urban area, water or none of the above using sample statistics of sections of images. The sample statistics include averages, correlation and absolute value, all of which are very easy to compute and the results are compared against an empirically determined threshold. Geometry is not examined and the algorithm works very well when only one of the four objects are in the scanning scene. It remains to be determined how such algorithms, which depend on averages of the gross image, work when there is an overlap with other boundary objects.

In this report we are concerned principally with the second of the categories of cartographically important objects, namely the discrete man-made ones. This class of objects lend themselves to enhancement; and the recognition of them is enormously benefited by various types of digital image preprocessing such as linear transforms, edge detectors, local contrast changes and the use of a priori knowledge of the geometrical characteristics of objects in question. The main emphasis in our work was to develop a set of working tools which could thus be incorporated into a transportable algorithm that would work on <u>actual</u> images. To illustrate the principles and to develop a concrete example, we have placed most of our efforts

on bridge-over water recognition. The objective was to have the algorithm be effective regardless of the other objects in the field, even those that superficially might resemble a bridge. In order to do this we developed and refined a set of digital preprocessing techniques such as edge detectors, image smoothers, straight-line transforms (quantized Hough and other variants), thresholding techniques and variations on the medial axis transform. In addition, we built in the a priori knowledge of bridge characteristics including the global environment. The actual process of bridge identification then consists of applying the preprocessing operations in sequence and completed with a decision if all tests are met. In a sense, this procedure can be viewed as a syntactical approach when the basic language primitives are the individual operations (edge detection, Hough transform, thresholding, etc.).

While the main thrust of the specific end product of this work has been in bridge detection, we have not been unaware of applicability of our methods to other objects such as airport runways, roads and industrial sites. We have made some inroads to the recognition of these objects as well. In addition, we have examined certain aspects of the coding of line drawings since it was felt that many of the man-made objects can be skeletonized to line drawings and that the coding process can be helpful in their identification.

In the chapters that follow, we present a condensation of the effort of the past year. The report is organized in a way to allow the reader to examine separately the models used, the

individual routines and algorithms, and their synthesis in an actual application. Standard FORTRAN code is included in the Appendix and a tape has already been provided.

CHAPTER II

APPROACHES TO AUTOMATIC CARTOGRAPHIC IMAGE CLASSIFICATION

Background

Classification of a cartographic image is used here to mean the assignment of a class label to the image or a subimage. The label would denote a standard cartographic feature--a bridge, road, river, etc.--and in general would identify a set disjoint from those corresponding to other labels. The image (or subimage--but henceforth all will be called images) is thus a pattern and a number of pattern recognition techniques are potentially applicable. The simplest approach is template matching, which compares templates of prototypical cartographic objects to the image being examined. The template that is closest (as determined with a suitable metric) is declared the class of the image. Template matching is computationally easy and fast; it is, however, strongly dependent on the orientation and scale of the image; the quality of the image can also affect the match.

That sensitivity gave rise to the use of features that jointly characterize the pattern and that exhibit less variation as the image departs from the prototype or is corrupted by noise. Features are measurements made on the pattern and

are chosen to be both easy to extract and effective in separating the classes. A set of n features defines an n-dimensional feature space within which we want the patterns to be well separated.

An image in analog form (e.g., a continuous-tone photograph) must undergo several preprocessing operations before it is in a form suitable for feature extraction. An overall block diagram illustrating the major processes that are usually implemented in a modern digital image processing system is shown in Figure 2.1, and a general description of each of these processes follows. The first process usually applied to an input picture is that of digitization, in which this original picture is converted into another two-dimensional representation that is suitable for digital computer processing. Two examples of such pre-processing are quantization (i.e., analog-to-digital conversion) in both space and amplitude and down-sampling (i.e., one way of reducing the number of bits used in representing the source). The second process is often that of edge detection and thresholding, whereby the digitized image is examined for large changes in intensity followed by a thresholding operation that retains only those edges that are likely to be significant in subsequent processing. Next is usually an image segmentation process, in which certain characteristics (e.g., texture) are utilized to subdivide the image into distinct nearly homogeneous regions. These resulting regions could then be encoded (as described below) to remove the redundant information within them, while still maintaining their essential attributes. Feature

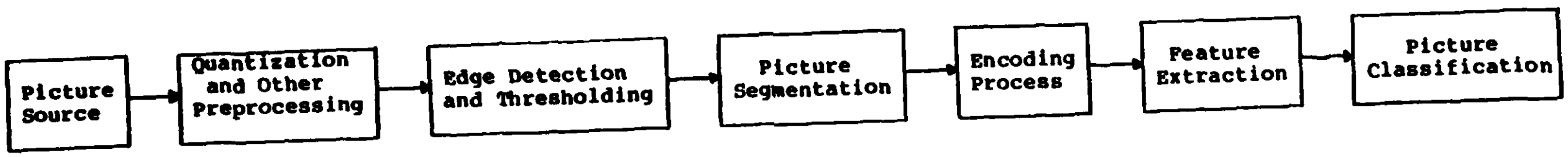


Figure 2.1 An Image Processing System

extraction follows; typical features in cartographic images include straight lines, regularity or periodicity, aspect ratios, and relative sizes. Finally, the features are used in a classifier that may evaluate a linear or nonlinear function of the features and compare it to a threshold or use the features sequentially, stopping when a decision can be made. Many classifier designs exist and tradeoffs between feature complexity and classifier complexity are inevitable.

Each of the major processes that were briefly discussed above (in reference to Figure 2.1 will be discussed in greater detail in the subsequent sections of this report. However, it is first important to remember what is the ultimate objective of the system shown in Figure 2.1: to extract and correctly identify particular objects that are of interest to the user, given an input picture. With this in mind, it is necessary then to analyze the list of objects (or targets) of interest to the user and attempt to categorize them according to their obvious features. Such an analysis is performed in the following section.

Preliminary Target Classification

In this study several targets of high interest--bridges and airports--have been identified for focused investigation. Table 2.1 categorizes a wide range of cartographic targets in terms of their origin, either natural or man-made. Since the target identification process is facilitated by identifying cues or component attributes, and since it is desired to make

Table 2.1

High-Interest Target Set

Natural	Man-Made
• Rivers and Streams	• Isolated Buildings
• Unpaved Roads and Trails	• Storage Tanks
• Rapids and Falls	• Quarry or Borrow Pit
• Shoreline	• Tunnel Entrance
• Large Rivers	• Canals
• Lakes	• Dual Highways
• Forests	• Primary Roads
• Scrub	• Secondary Roads
• Marsh and Swamp	• Train-Tracks
	• Transmission Line
	• Pipe Line
	• Levee
	• Dam
	• Bridge
	• Airport
	• Orchard and Vineyard
	• Urban Area
	• Suburban Area
	• Industrial Area
	• Railroad Area
	• Cemetery

decisions that distinguish between different classes of targets, it is desirable to use unique sets of attributes to aid in the decision process, each target class being associated with several attributes unique to its character. If, for example, the attribute "long, thin, parallel strip" is a co-attribute of ten of the high-interest targets, then it is of only minimal value since it does not uniquely identify a particular member of the set of targets. Of course, there are other objects and shapes that may appear in aerial images that do not appear in the list of 31 high-interest targets. These may be improperly identified as one of the high-interest candidates if the attribute set is sufficiently broad. The identification of similar characteristics and elementary cues can serve to structure the problem and to help eliminate the selection of those characteristics or attributes that lend little to the decision process.

In this brief section the members of the high-interest target set will be considered in terms of their similarities and differences by classifying them into subsets. Generally the classification of objects into natural and man-made is done to identify those targets that exhibit smooth lines (man-made) versus those that exhibit irregular lines (natural objects). (Freeman makes note of this general characteristic in his review article on computer processing of line drawings). A river will exhibit, for example, a shoreline that randomly curves and forks. On the other hand, a man-made canal will generally be built with well known geometrical shapes and will have smooth sides.

One of the factors that precipitated this brief look at the target classes was the inclusion of several very similar road or road-like structures in the table. If these are indeed to be distinguished from one another, then specific characteristics or combinations of them that set them apart must be identified.

In the table, for example, there are seven candidates that exhibit characteristics that are road-like:

- Unpaved Roads and Trails
- Dual Highways
- Primary Roads
- Secondary Roads
- Train Tracks
- Transmission Lines
- Pipe Lines

All appear from the air as smooth-sided ribbons of uniform luminance or texture; all exhibit parallel sides. Often all of the seven are built using long straight segments. All include intersections or forks that allow interconnections of two or more of their numbers. If an edge-detection algorithm is used to create a line drawing depicting these seven objects, what is to distinguish them? Perhaps the specific dimensions can be used in some cases, but no clear distinction is obvious based only on this parameter. In some cases other attributes may be available to assist in separating the targets within a class. In the case of road-like structures, paved roads may be differentiated from transmission lines, pipelines, and railroad rights of way based on the textured properties, both within the parallel line structure and adjacent to it.

In summary, all exhibit these characteristis:

- High Aspect Ratio (L/W)
- Limited Width
- Uniform Width
- Smooth Curves or Straight-Sided. May include sharp angles or intersects
- Intersected by other road-like structure
- Not Isolated. Usually continues off image
- Constrasting Surface with respect to surroundings

and these characteristics do not appear in any of the other targets (with the exception of airports and urban and suburban areas that will be classified here as "complex" targets, as will be defined later). This suggests that, at least in the case of road and road-like structures, a multi-tiered decision tree could be applied to structure the identification problem. Figure 2.2 shows a portion of the decision tree. Only the branches associated with the road-like geometry have been fully traced in the Figure. At the upper level are all target geometries. Envisioned at the next subsequent decision level are several algorithms that test the candidate image for membership in general classes that include characteristics like ribbon pattern (parallel, equally spaced lines of extended length), isolated building structure (relatively small, smooth-sided polygon of uniform luminance or uniform texture), water target (uniform luminance, constrasting with surroundings, relatively large surface area), and so on. (The example characteristics proposed here are only for illustrative purposes. As the targets are each studied in greater detail the specific attribute necessary to perform the decision process will be carefully developed.) At the next level of decision-making in the

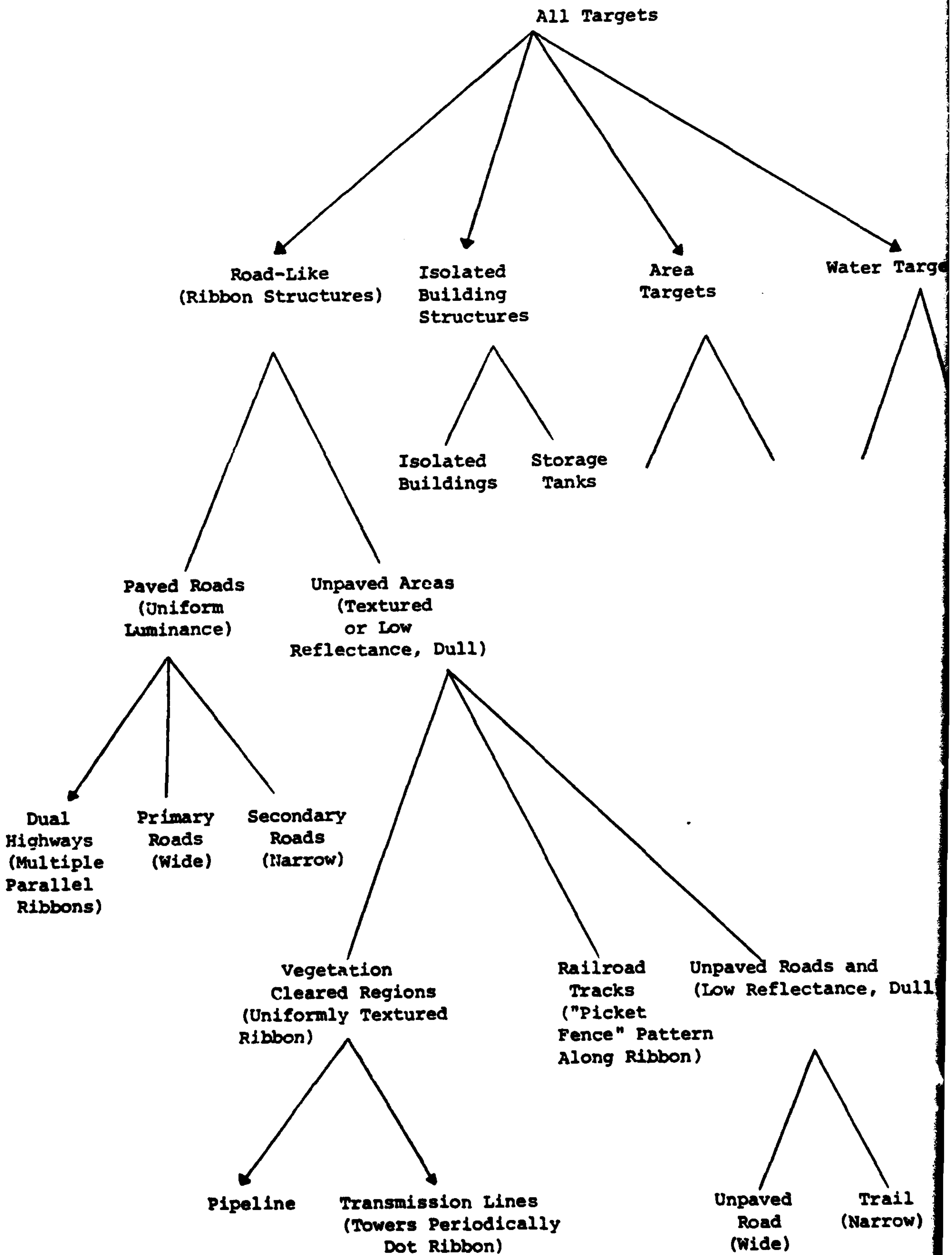


Figure 2.2 Decision-Tree Approach to Target Identification

decision-tree-branch representing the ribbon structure, the texture may be tested to determine if the candidate is of uniform luminance (paved roads), or if it is dull, or of low reflectance or textured (pipelines, transmission lines, unpaved roads and trails). At the next level the structure may be classified further depending upon the specific structure of the logic tree. In the case of the branch depicting roads, the key parameter that separates dual highways, primary, and secondary roads is the width and number of parallel ribbons that make up the image.

The decision tree described in Figure 2.2 invokes the notion of a sequential logic process. That is to say, the lower level tests and decisions in the tree are carried out only as necessary as the branches and branch points of the tree are traced out to a tip of the tree. Such a process could be implemented in a modular format using software modules to carry out each of the decision-related tests. These modules would be called upon to perform their specific testing operations on an as-required basis by an executive function that would react to the current position in the logic tree. Such a concept would be more efficient than one that would be designed to carry out all of the test operations in parallel. In the latter case the system would simultaneously perform all of the operations indicated in the examples in the logic tree irrespective of the results of other tests that may be performed. In such a decision structure the results of each of the tests could be placed in a feature vector much like those indicated in Table 2.2. In the Table a number of example target characteristics

Table 2.2
Typical Feature Vectors

	Water Associated (Water as Cue)	Line-Like (Smooth)	Large Area-Like	Complex Assembly	Small Area Targets-Regular	Water Contained Inside Periphery	Line-Like (Short)	Line-Like (Long)	...
Rivers & Streams	1	1	1	0	0	1	0	1	
Unpaved Roads	0	1	1	0	0	0	0	1	
Trails	0	1	0	0	0	0	0	1	
Rapids & Falls	1	0	0	0	0	0	0	0	
Shoreline	1	0	0	0	0	0	0	0	
Large Rivers	1	0	1	0	0	1	0	0	
Lakes	1	0	1	0	0	1	0	0	
Forests	0	0	1	0	0	0	0	0	...
Scrub	0	0	1	0	0	0	0	0	
Marsh & Swamp	0	0	1	0	0	0	0	0	
Isolated Bldgs	0	0	0	0	1	0	0	0	
Storage Tanks	0	0	0	0	1	0	0	0	
Quarry or Borrow Pit	0	0	1	0	0	0	0	0	
Tunnel Entrance	0	0	0	0	0	0	0	0	
Canals	1	1	1	0	1	1	1	0	...
Dual Hwys	0	1	1	0	0	0	0	1	
Primary Roads	0	1	1	0	0	0	0	1	
Secondary Roads	0	1	1	0	0	0	0	1	
Train Tracks	0	1	1	0	0	0	0	1	
Transmission Lines	0	1	1	0	0	0	0	1	
Pipe Lines	0	1	1	0	0	0	0	1	
Levees	1	0	0	0	0	0	0	0	
Dam	1	1	0	0	0	0	0	0	...
Bridges	1	0	0	0	0	0	0	0	
Airport	0	0	1	1	0	0	0	0	
Orchard & Vine Yard	0	0	1	0	0	0	0	0	
Urban Area	0	0	1	1	0	0	0	0	
Suburban Area	0	0	1	1	0	0	0	0	
Industrial Area	0	0	1	1	0	0	0	0	...
Railroad Area	0	0	1	1	0	0	0	0	
Cemetery	0	0	1	0	0	0	0	0	

1 = Yes (definitely)
0 = No (but maybe)
x = ?

are listed across the top. They are not necessarily the most practicable set for cartographic purposes. Each of the target candidates is listed along the side of the matrix. When a target exhibits a particular attribute a "one" is entered in that attribute's position in the vector, and when the target does not exhibit the attribute, a "zero" is entered in the attribute's position. The Table is minimally constructed (i.e., a minimum number of attributes have been defined) when all of the target candidates exhibit unique decision vectors. When the decision table is completely constructed then tests of candidate targets can be made, and the identification process reduces to one of matching the candidate target's decision vector with those entered in the Table. As additional attributes are identified and made available that result in unique decision vectors, the probability of a correct decision should increase since more information is being used in making the final identification decision.

One exercise that was carried out during consideration of the entire suggested target set of 31 candidates was a grouping along the lines of eight major categories. These major categories could lend themselves to the logic tree format discussed here. Within each major grouping a second-level decision would then be applied to further separate the target candidates. Although the specific characteristics necessary to carry this process to completion are forthcoming, the process is a natural extension of that described earlier.

The seven major target categories and their members are

I. Road and Road-Like (Ribbon)

- Dual Highways
- Primary Roads
- Secondary Roads
- Railroad Tracks
- Transmission Lines
- Pipelines
- Trails
- Unpaved Roads

II. Simple Large Scale "Area" Targets (Irregular)

- Cemetery
- Orchards and Vineyards
- Quarry or Borrow Pit
- Marsh and Swamp
- Scrub
- Forests

III. Small "Area" Targets (Regular)

- Isolated Buildings
- Storage Tanks

IV. Complex (Aggregate) Targets--Large Scale

- Airport
- Urban Area
- Suburban Area
- Industrial Area
- Railroad (Switchyard) area

V. Water-Bearing

- Rivers
- Large Rivers
- Streams
- Lakes
- Canals

VI. Appurtenances of Water-Bearing Bodies (cued by bodies of water)

- Levees
- Dams
- Bridges (road or railroad may also cue)
- Shoreline
- Rapids and falls

VII. Miscellaneous Category

- Tunnel Entrances

The discussion above illustrates the interaction that must occur between an intuitive approach (i.e., one that would model the decision procedure after that followed by a human photo interpreter) and a formal one that relies on the structure of the data. Formal methods nevertheless require a data set that will be examined for structure; the features that are initially measured must represent the designer's estimate of discriminatory power. Statistical methods provide guidance in pruning the candidate feature set when a large data set exists, but the small data base used here makes the decision-tree method seem reasonable. The next chapter presents some tools used for extracting those features.

CHAPTER III

ROUTINES FOR BRIDGE PATTERN DETECTION

The segment of the project discussed in the following has as its goal the generation of a small set of structures from an image that are "potential bridges." These bridge candidates are sets of parallel lines satisfying certain conditions; namely, pairs of lines that are long and relatively close together. Each possible "bridge" is then subject to a set of tests that result in the structure being labelled "bridge" or "not a bridge". The process involves four primary steps:

(1) Pre-processing (if desired or necessary to down-sample, reduce noise or both).

(2) Edge detection

(3) Recognition of long, parallel lines from the edge-detected field.

(4) Testing the resulting lines for "bridge" or "non-bridge" conditions.

Each of these steps is discussed in turn.

Pre-Processing Techniques

Two pre-processing routines have been developed: image down-sampling and image noise reduction. The potential desirability of these two procedures are clear. For example, the original images being processed have been digitized to 256 pixels/inch. The image scale, however, may be such that the

structures to be detected are on the order of millimeters or more. Thus, a large computational advantage may be gained with minimal relevant information loss by down-sampling the image to, say, 64 pixels/inch (a 4-to-1 down-sampling). The benefit is a factor of 16 reduction in the number of pixels to be processed while (as will be seen) retaining essentially all relevant detail.

The algorithm of Appendix 3.1 allows variable ratio down sampling in two manners: first, by simply extracting every n-th pixel in every n-th row (giving an n-to-1 down-sampling); or second by collapsing every n-by-n square of pixels into a single pixel with a value equal to the average of the n^2 pixels. The first method is computationally much faster and appears to be equally effective in the images being used. Almost all of the examples shown in the following are the result of processing sections of a 4-to-1 down-sampled image resulting from method one. Figure 3.1 is the 4-to-1 down-sampled image section containing a narrow bridge.

The algorithm of Appendix 3.2 is that used for image noise reduction. One of two types of noise reduction may be performed: (a) eliminating isolated noise "spikes" via thresholding; or (b) image smoothing using one of three convolutional masks.

Noise spike elimination is performed as follows: the value of each pixel is compared to the average of its eight nearest neighbors. If the absolute value of the difference between the pixel value and the average is greater than a threshold, the

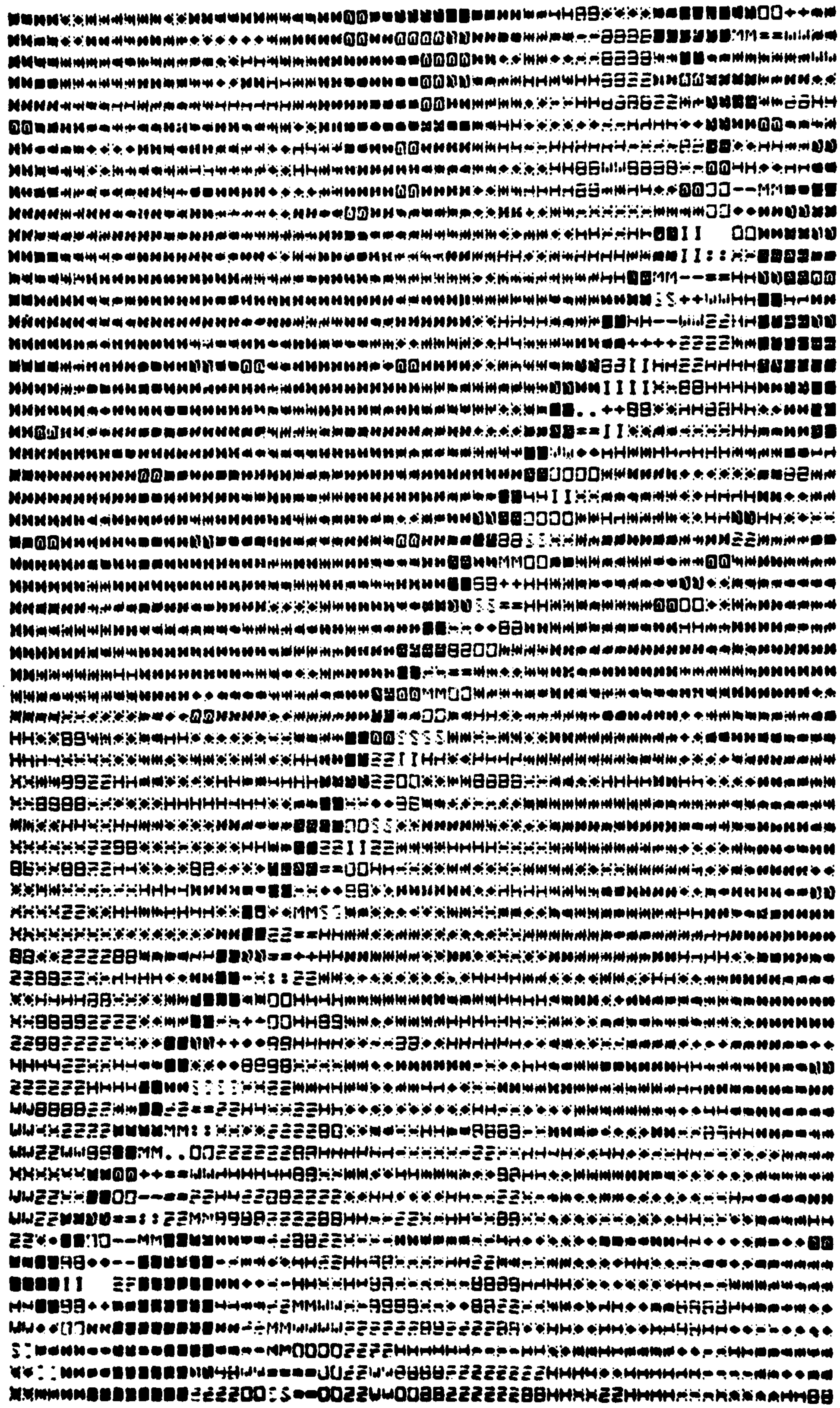


Figure 3.1. Downsampled Digitized Image Containing a Bridge

pixel is set equal to the average. The threshold level may be dynamically varied over the image by setting it equal to a specified number of standard deviations above the 8-neighbor average. Thus, regions of "smoothness" are compared to a low threshold while rapidly varying regions are compared to a high threshold.

Image smoothing is achieved by specifying one of three 3 x 3 convolutional masks providing a range of smoothing. The three masks are:

Mask 1: $\frac{1}{9} \cdot \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline\end{array}$ Mask 2: $\frac{1}{10} \cdot \begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 2 & 1 \\ \hline 1 & 1 & 1 \\ \hline\end{array}$ Mask 3: $\frac{1}{16} \cdot \begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline\end{array}$

As can be seen, the masks differ in the weighting given to pixels nearer the central pixel. The convolutions are all normalized so that the average image brightness is unchanged. Figure 3.2 is an example of smoothing using Mask 1. As is expected, the image of the bridge (corresponding to the top half of Figure 3.1) is broadened and its edge contrast reduced.

Edge Detection

A review of the literature reveals a large number of edge detection algorithms for digital images, most falling into one of two categories: template-matching operators [21], [31], and differential operators [11], [14], [33]. The latter group includes both 2 x 2 pixel operators ("Robert's" operator) and the 3 x 3 pixel operators ("Sobel" and "Prewitt" operators). Comparisons of these techniques [1], [10], [13] in terms of performance

Figure 3.2. Smoothed Version of Upper Half of Figure 3.1

and computational simplicity suggest that the group of 3 x 3 differential operators may be preferred for the current task. Thus, it is to this group that attention was directed.

In general, the 3 x 3 differential operators are of the form:

$$\underline{F} = \begin{array}{|c|c|c|} \hline 1 & W & 1 \\ \hline 0 & 0 & 0 \\ \hline -1 & -W & -1 \\ \hline \end{array} \qquad \underline{G} = \begin{array}{|c|c|c|} \hline 1 & 0 & -1 \\ \hline W & 0 & -W \\ \hline 1 & 0 & -1 \\ \hline \end{array}$$

With W = 1, the operator is known as the "Prewitt" or "smoothed gradient" operator. With W = 2, it is the "Sobel" operator. $\underline{F}$ and $\underline{G}$ are simultaneously convolved with the digitized image and an edge is judged to be present if:

$$(\underline{F} \cdot \underline{B})^2 + (\underline{G} \cdot \underline{B})^2 \geq A$$

where A is a pre-set threshold value, and B is the 3 x 3 sub-image currently being tested. An alternative threshold expression compares the sum of the absolute values rather than the sum of the squares, for computational efficiency.

Two potential "improvements" in the performance of the 3 x 3 differential operators have been proposed by Frei and Chen [14]. The first modification suggests using $W = \sqrt{2}$. The result is an "isotropic" operator: that is, one in which an edge is equally likely to be detected regardless of its angular orientation. To see this, consider the detection of an arbitrary edge such as in Figure 3.3 located at 90° and 45°.

$$\underline{A} = \begin{array}{ccccc} 2 & 2 & 1 & 0 & 0 \\ 2 & 2 & 1 & 0 & 0 \\ 2 & 2 & 1 & 0 & 0 \\ 2 & 2 & 1 & 0 & 0 \\ 2 & 2 & 1 & 0 & 0 \end{array} \qquad \underline{B} = \begin{array}{ccccc} 2 & 2 & 2 & 2 & 1 \\ 2 & 2 & 2 & 1 & 0 \\ 2 & 2 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{array}$$

Figure 3.3. Detection of an Arbitrary Edge

Centering the 3 x 3 operator over the central 3 x 3 images (in the square) and letting W = 1 ("Prewitt") we get:

$$\underline{F}\cdot\underline{A} = 1\cdot(2+1+0) + 0\cdot(2+1+0) + (-1)\cdot(2+1+0) = 0$$

$$\underline{G}\cdot\underline{A} = 1\cdot(2+2+2) + 0\cdot(1+1+1) + (-1)\cdot(0+0+0) = 6$$

Thus: $(\underline{F}\cdot\underline{A})^2 + (\underline{G}\cdot\underline{A})^2 = 0^2+6^2 = 36$

For $\underline{B}$: $\underline{F}\cdot\underline{B} = 1\cdot(2+2+1) + 0\cdot(2+1+0) + (-1)\cdot(1+0+0) = 4$

$$\underline{G}\cdot\underline{B} = 1\cdot(2+2+1) + 0\cdot(2+1+0) + (-1).(1+0+0) = 4$$

Thus: $(\underline{F}\cdot\underline{B}^2 + (\underline{G}\cdot\underline{B})^2 = 4^2+4^2 = 32$

That is, the identical rotated 45° results in a lower edge "magnitude" and, all else equal, a poorer chance of exceeding any specific threshold. Similarly, with W = 2 ("Sobel") we obtain:

$$(\underline{F}\cdot\underline{A})^2 + (\underline{G}\cdot\underline{A})^2 = 64 \quad (90°)$$

$$(\underline{F}\cdot\underline{B})^2 + (G\cdot B)^2 = 72 \quad (45°)$$

and again the 45° edge results in a higher value (72 vs 64). If, instead, we let W = $\sqrt{2}$ ("isotropic") it is observed that:

$$(\underline{F}\cdot\underline{A})^2 + (\underline{G}\cdot\underline{A})^2 = 36.62$$

$$(\underline{F}\cdot\underline{B})^2 + (\underline{G}\cdot\underline{B})^2 = 36.62$$

i.e., an equal value results from the edge operator at the different angles.

The second potential improvement deals with the manner in which the edge thresholding is performed. Suppose the 3 x 3 subimage to be subjected to the edge operator is considered to be a vector in a 2-dimensional space; the basis vectors of this space are the "edge" vector (i.e., F + G)* and a vector representing "all else." This is depicted in Figure 3.4. Traditionally, the thresholding is performed by determining the edge space component of B (i.e., the dot product of B with the edge vector, or edge operator) and comparing it to a threshold value A. Now consider the situations illustrated by Figure 3.5.

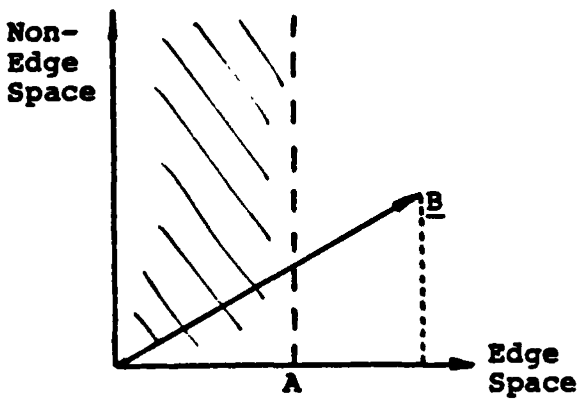


Figure 3.4. Thresholding in Edge Space: Case 1

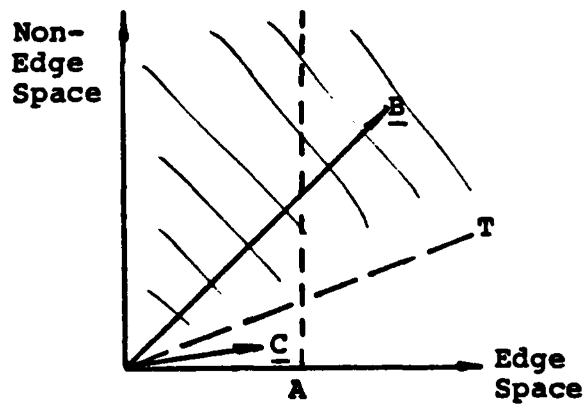


Figure 3.5. Thresholding in Edge Space: Case 2

---

*Actually, the "edge space" defined by Frei & Chen had 2 additional components (H and I, at right) to F and G. In practice, including H and I added little to the operator's performance.

$$\underline{H} = \begin{bmatrix} 0 & -1 & \sqrt{2} \\ 1 & 0 & -1 \\ -\sqrt{2} & 1 & 0 \end{bmatrix} \qquad \underline{I} = \begin{bmatrix} \sqrt{2} & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & [illegible] \end{bmatrix}$$

Here, the projection of $\underline{B}$ onto the edge space (its edge component) exceeds the threshold A, although $\underline{B}$ actually appears to be closer to the non-edge space. Alternatively, $\underline{C}$ is very close to the edge space but, due to its small magnitude, would not be called an edge point since its dot product with the edge space is less than A. Neither case would have occurred if an "angular threshold" had been used; that is, if thresholding had been performed not on the magnitudes of $\underline{B}$ or $\underline{C}$, but on the angle Θ between the vector and the edge subspace.

If: $$X \stackrel{\Delta}{=} (\underline{B}\cdot\underline{F})^2 + (\underline{B}\cdot\underline{G})^2$$

is the magnitude of the edge space component of the image vector $\underline{B}$, then θ is calculated as follows:

$$\cos\Theta = \sqrt{X}/\sqrt{(\underline{\bar{B}}-\underline{B}).(\underline{\bar{B}}\ \underline{B})} \rightarrow \Theta = \arccos(X/(\underline{B}-\underline{\bar{B}}))^{1/2}$$

If Θ' is the selected threshold, then we will call $\underline{B}$ an edge point if:

$$\arccos \frac{(\underline{B}\cdot\underline{F})^2 + (\underline{B}\cdot\underline{G})^2}{(\underline{B}-\underline{B})^2} \quad 1/2 \quad \equiv T \leq \Theta'$$

or equivalently, since the cosine is monotonically decreasing between 0 and π

$$T \geq \cos^2 \Theta' \equiv A'$$

Comparing to the previous threshold expression, the net effect is that the original test statistic, $(\underline{B}\cdot\underline{F})^2 + (\underline{B}\cdot\underline{G})^2$, is divided by the magnitude of the edge "signal" under consideration, and compared to a new threshold A'. From an intuitive viewpoint,

this modified threshold tests for the "closeness of fit" to being an edge, rather than "strength" of the possible edge signal.

Using either threshold, the orientation, γ , of a detected edge point is given by:

$$\gamma = \arctan [(\underline{G} \cdot \underline{B})/(\underline{B} \cdot \underline{B})]$$

A computer subroutine has been developed to perform the edge detection operation by convolving the 3 x 3 operator with an input image (Appendix 3.3). The subroutine calls the edge operator (Appendix 3.3) for each pixel I and returns for each pixel judged to be an edge its magnitude and orientation. The value of W (1 = "Prewitt", 2 = "Sobel", SQRT(2) = "isotropic," etc.) is specified, as well as the type of threshold (magnitude or angular) and threshold level.

Figures 3.6 and 3.7 provide examples of subimages subjected to edge detection using both types of thresholding. Thresholds were adjusted until both methods generated equal numbers of edge points. As can be seen, the angular threshold detected many low contrast edges in the water region which, although really existing, are irrelevant. Thus, since the current task involves detection of structures which may be of relatively high contrast, the higher sensitivity of the angular threshold to low-contrast edges may be wasted (or worse, interfering). The method of choice for this project, however, remains to be determined from experience.

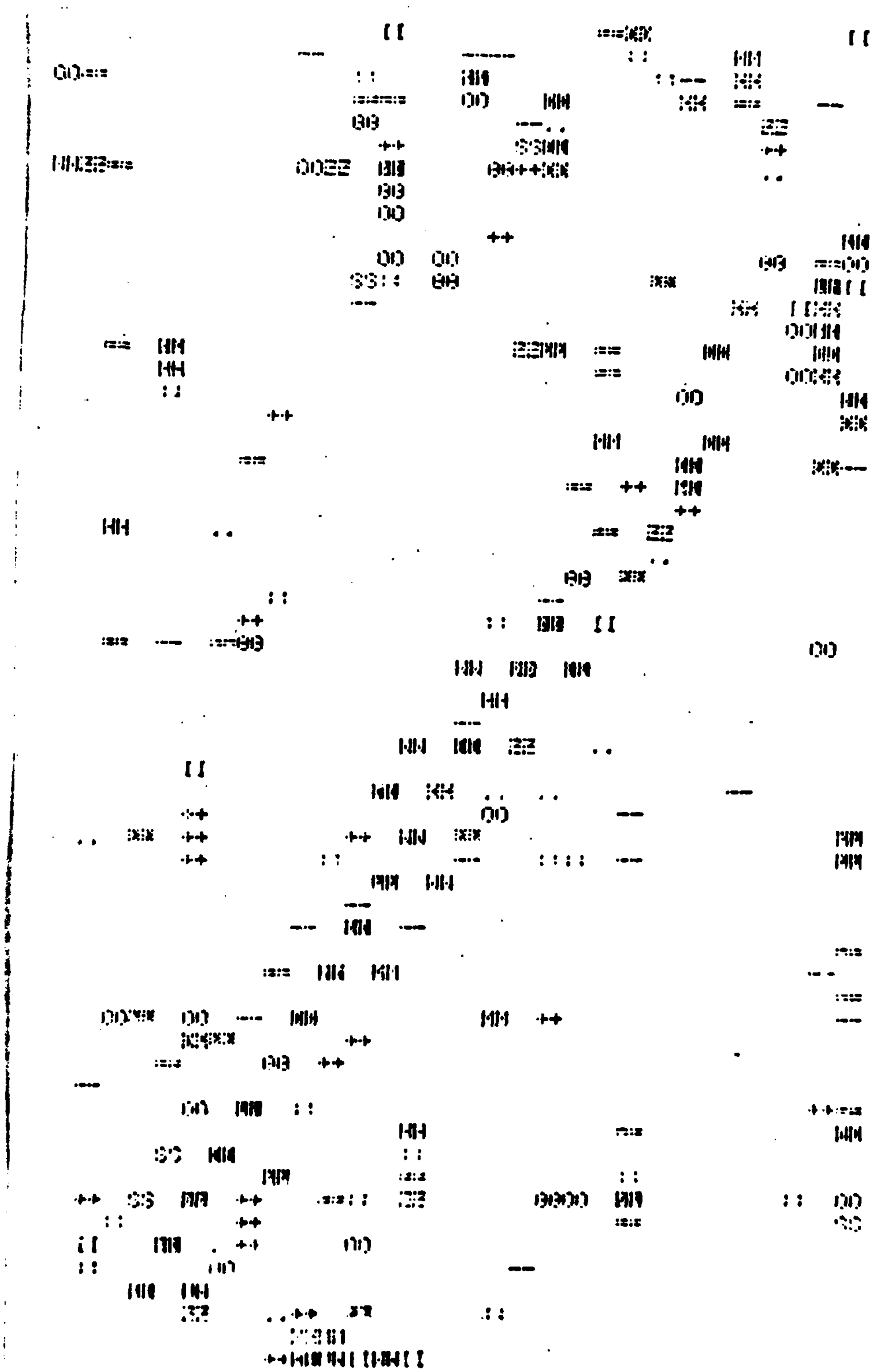


Figure 3.6. Edge Detection Using Angular Thresholding (Image segment corresponds to Fig. 3.1; threshold = 0.55 radians.)

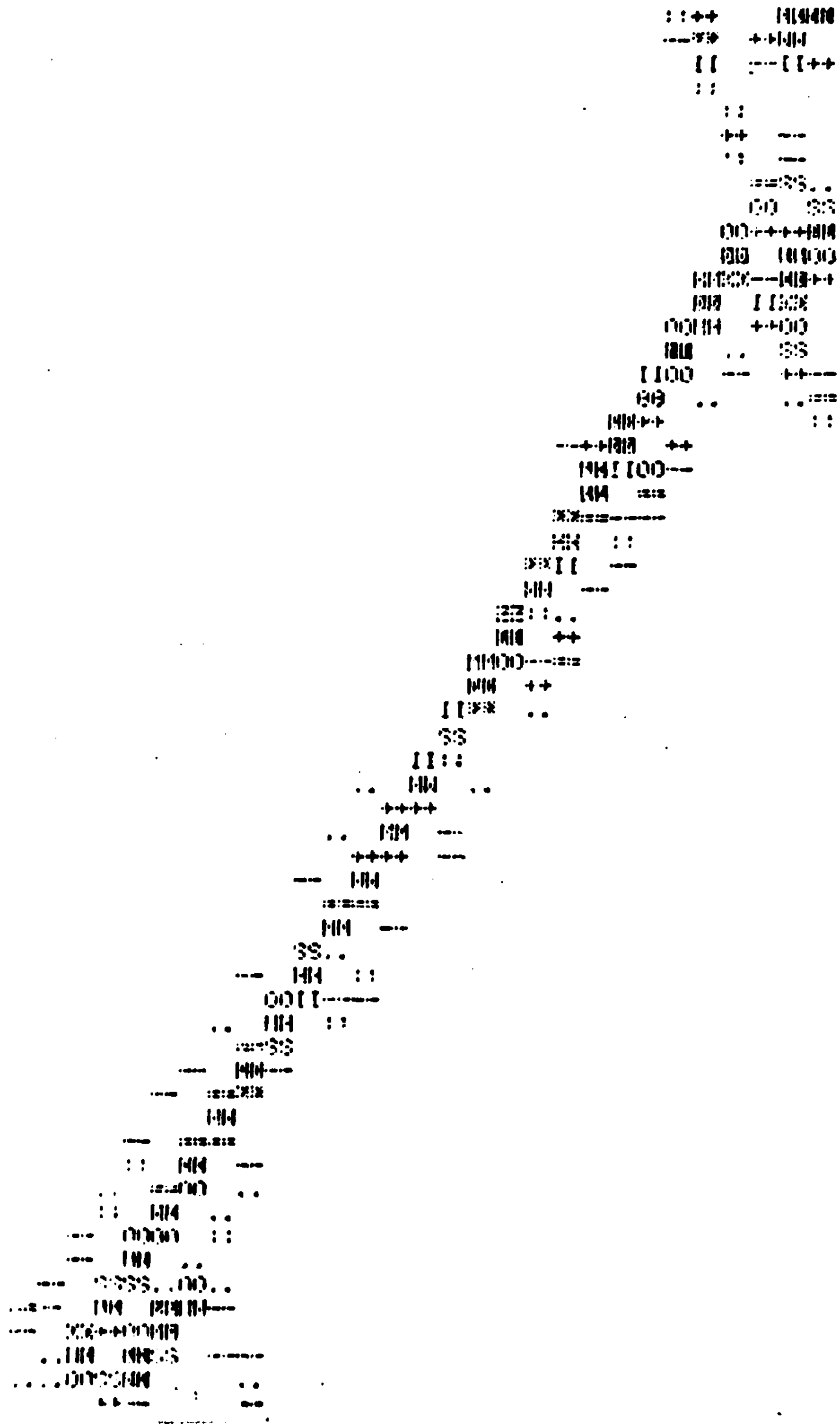


Figure 3.7. Edge Detection Using Magnitude Thresholding (Image segment corresponds to Fig. 3.1; threshold = 500; 128 detected edge points.)

Line Recognition

The output of an edge detection algorithm such as described above is, in general, a set of discrete, disconnected edge segments, often appearing to be randomly placed. An example is shown in Figure 3.8. The next test is usually to use a "linking" or "line building" algorithm. These algorithms generally look for line segments that fall within a given tolerance of distance and orientation of each other. When such segments are found, they are linked, forming a single longer line segment. This process continues until all long lines (if present) are built up. All segments which are below some threshold value in length are dropped.

The above process, however, can be time consuming and, for the task of (for example) detecting bridges may not even be necessary. In the present task, we are not so much interested in line-building as in the specific question "are there long lines present in the image?" and, if so, "are there parallel lines whose lengths are much greater than their separation?". These questions arise since bridges are, of course, composed of long, close parallel lines. These types of questions may be answered using a Hough transform.

Basically, a Hough transform operates on a set of predetermined feature points in the image (or X-Y) space. The set of edge points resulting from an edge detection operation is such a set of feature points. The Hough transform uses these points to generate a set of points in rho-theta space, where the rho and theta values are coordinates of a line in X-Y space. That is, it is a line-to-point transformation.

Figure 3.8. Image With Apparently Random Structure (Lower half is edge-detected image.)

To explain this transformation, assume that some line L exists in the image (X-Y) space). This line can be described by two coordinates: rho(R), the perpendicular distance of the line to the origin (which may be selected, for example, to be the lower left corner of the image), and theta (Θ), the angle of the perpendicular with the X-axis (Figure 3.9). Now, let B be an edge point of L detected by an edge operation. B is thus a "feature point". An infinite number of lines may pass through this point. Suppose, however, that we quantize the angles of the candidate lines into, say, eight values between 0 and 180° (22.5° increments). Thus, we allow one of eight

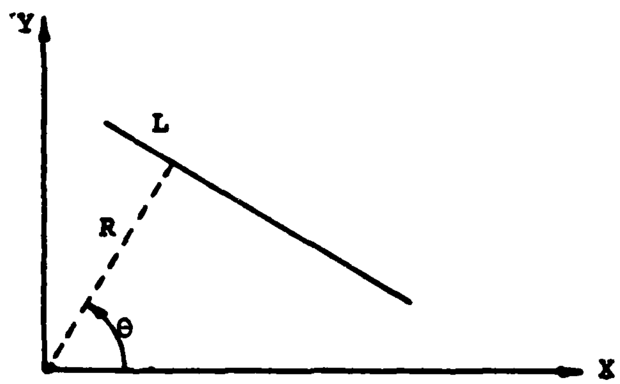


Figure 3.9. Hough Transform Definition

possible lines to pass through B (Figure 3.10a). We now plot the coordinates of each of these eight lines in (R-Θ) space (Figure 3.10b). In general, the result is a curve as in Figure 3.10b.

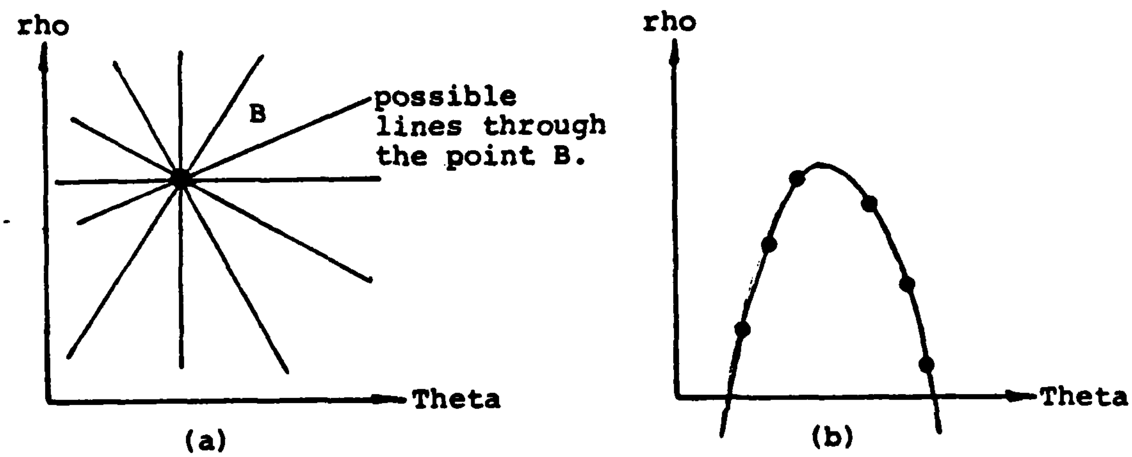


Figure 3.10. Hough Transform Space

Now, suppose we have quantized rho into, say, n levels so that (R-Θ) space is represented by a 2-dimensional (R-Θ) matrix Figure 3.11). We now perform the plotting procedure for each of the 8 possible lines by incrementing (by 1) the appropriate cell in the (R-Θ) matrix. An identical process is performed for every feature point in X-Y space (i.e., every detected edge point). The final result is a matrix (in which each element defines a particular (R-Θ) pair) whose entries are

equal to the number of times each cell was incremented. That is, a cell with a value of 20 implies that 20 edge points were detected each of which has one of its eight possible lines possessing those (R-Θ) coordinates. Since, of course, all lines with

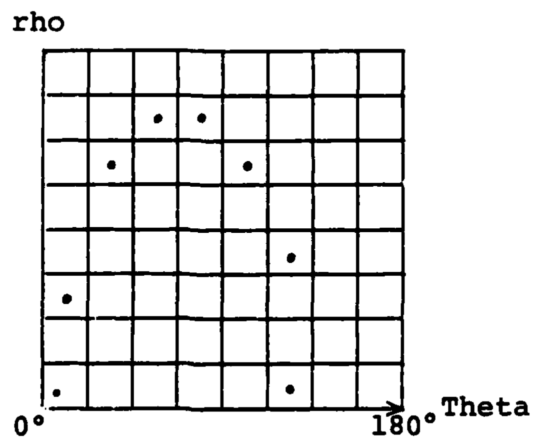


Figure 3.11. The R-Θ Matrix

the same coordinates are collinear, a (R-Θ) matrix element with a high value is the same as saying that a large number of feature (i.e., edge) points lay along the same line. Thus (R-Θ) elements with values above some threshold are judged to be lines (or edges) that exist in X-Y space.

Cells with the same value of Θ but differing R's represent parallel lines in the image. Cells exceeding the threshhold that have the same Θ with R's that are close together define image lines that are long, parallel and close together. These are our "potential bridges" (Figure 3.12).

Although more to the point, the above process is still calculation-intensive with fine quantization, even if the

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	1	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	1	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	1	1	0	0	0	1	0	0	0	0	0	0
10	0	0	0	0	0	1	0	0	0	0	0	0	0
11	0	0	1	0	0	1	0	0	0	0	0	0	0
12	0	0	0	1	0	2	1	0	0	0	0	0	0
13	0	0	0	1	1	4	0	0	0	0	0	0	0
14	0	0	0	3	11	4	0	0	0	0	0	0	0
15	0	0	1	4	18	5	1	0	0	0	0	0	0
16	0	0	0	4	15	5	3	0	0	0	0	1	0
17	0	0	0	2	6	3	3	0	1	1	0	1	0
18	0	0	0	3	0	0	1	6	1	0	0	0	0
19	0	0	0	0	0	0	0	2	0	0	0	0	0
20	0	0	0	0	0	0	0	0	1	0	0	0	0
21	0	0	0	0	0	0	0	0	1	0	0	0	0
22	0	0	0	0	0	0	0	0	2	0	1	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0

ANGLE DIVISION = .483077

Figure 3.12. Example of a Hough Transform

number of feature points is relatively small. However, if the feature points are generated by the edge detection mechanism previously described, then we are not using all of the information at hand. That is, we already have an estimate of the orientation of the line passing through each edge point: namely,

$$\gamma = \arctan[(\underline{F}\cdot\underline{B})/(\underline{G}\cdot\underline{B})]$$

where $\underline{F}$,$\underline{G}$, and $\underline{B}$ are as defined before. Thus, rather than calculating rho and theta for a number of possible lines (the number depending on the quantization of theta) and incrementing each of the appropriate (R-Θ) cells, we do it only for that line most likely to pass through each feature point (i.e., that with coordinates R,Θ) [15]. Since we already know Θ, we calculate rho:

$$R = X\cdot\cos\Theta + Y\cdot\sin\Theta$$

where X and Y are the image space coordinates of the feature point. The net result is a single addition calculation (of R) for each feature point to obtain the Hough transform, and a far more efficient way of generating the required "line existence" information.

A subroutine to perform the Hough transformation in this manner is given in Appendix 3.3. The program calls the edge operator for each pixel after which R and Θ are calculated and the appropriate cell incremented. Thus, this routine performs image edge detection and Hough transformation simultaneously.

Two weaknesses of the Hough transform become evident when it is applied to a large input image: first, that a large

amount of memory is required to store the (R-Θ) *matrix if fine* quantization of rho is desired; and second, that no information is provided about where in the image the lines exist (i.e., no end points are provided). These difficulties may be avoided by using the following scheme: divide the image into a number of small (for example 32 x 32 pixel) sub-images. Then operate on these subimages individually. The advantages are as follows:

(1) since the distances in X-Y space covered by the sub-images will be small, coarse quantization of rho will be sufficient to maintain fine spatial detail. (Quantization of Θ is already limited by the accuracy of the edge detection angle calculation.)

(2) again, since the spatial distances involved are relatively small, the question of "where are the line endpoints" is less important since it is limited to the range covered by subimage.

As will be described next, the final result of this scheme is a relatively small set of "bridge candidates" (long, close parallel lines). These may then be conveniently subjected to a series of tests or further analyses to determine its "bridge" or "non-bridge" status.

Figures 3.13 and 3.14 give Hough transforms and inverted Hough transforms following cell thresholding, for subimages using both types of edge detection described previously to generate feature points. In both cases, the bridge candidates clearly stand out and are easily isolated, although as before, there are fewer irrelevant entries resulting from the edge magnitude thresholding process.

Additional Analysis

Although work on this step has just been started, the procedure is relatively straightforward once the bridge candidates

Figure 3.13. Example of Hough and Inverse Hough Transforms with Highway. Hough (right) and inverse (below) based on upper part of Fig. 3.8. Boundaries have been extracted from cluttered image.

Theta

Rho												12
1	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	1	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
2	1	0	2	1	1	3	0	0	0	0	2	[illegible]
3	1	2	1	3	0	0	0	1	1	2	0	
4	1	1	0	1	0	0	0	0	0	0	1	
5	0	7	2	0	2	3	1	0	1	1	2	[illegible]
6	3	7	2	1	0	2	3	0	1	3	0	
7	2	4	1	6	1	1	0	0	0	0	0	
8	3	0	2	2	2	1	2	0	1	2	2	[illegible]
9	1	4	3	3	7	5	1	2	2	1	0	
10	0	3	0	2	1	5	3	3	2	3	3	[illegible]
11	3	1	3	8	5	8	5	2	1	2	4	
12	2	2	4	6	5	1	5	6	2	0	1	
13	0	0	2	4	3	9	2	3	4	4	2	
14	1	2	1	4	13	0	5	2	1	2	2	[illegible]
15	1	2	0	1	7	8	4	4	1	3	3	[illegible]
16	0	0	2	7	14	5	4	1	1	2	1	[illegible]
17	0	1	2	2	5	3	2	2	0	1	0	[illegible]
18	3	0	2	1	0	0	2	0	0	2	2	[illegible]
19	0	0	1	1	0	0	0	0	0	1	0	[illegible]
20	0	0	1	0	0	0	0	0	0	0	0	[illegible]
21	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	[illegible]	[illegible]	0	0	1	[illegible]	1	0
23	0	0	0	0	0	0	0	0	1	[illegible]	0	0
24	0	[illegible]	0	0	0	0	[illegible]	0	0	[illegible]	[illegible]	0

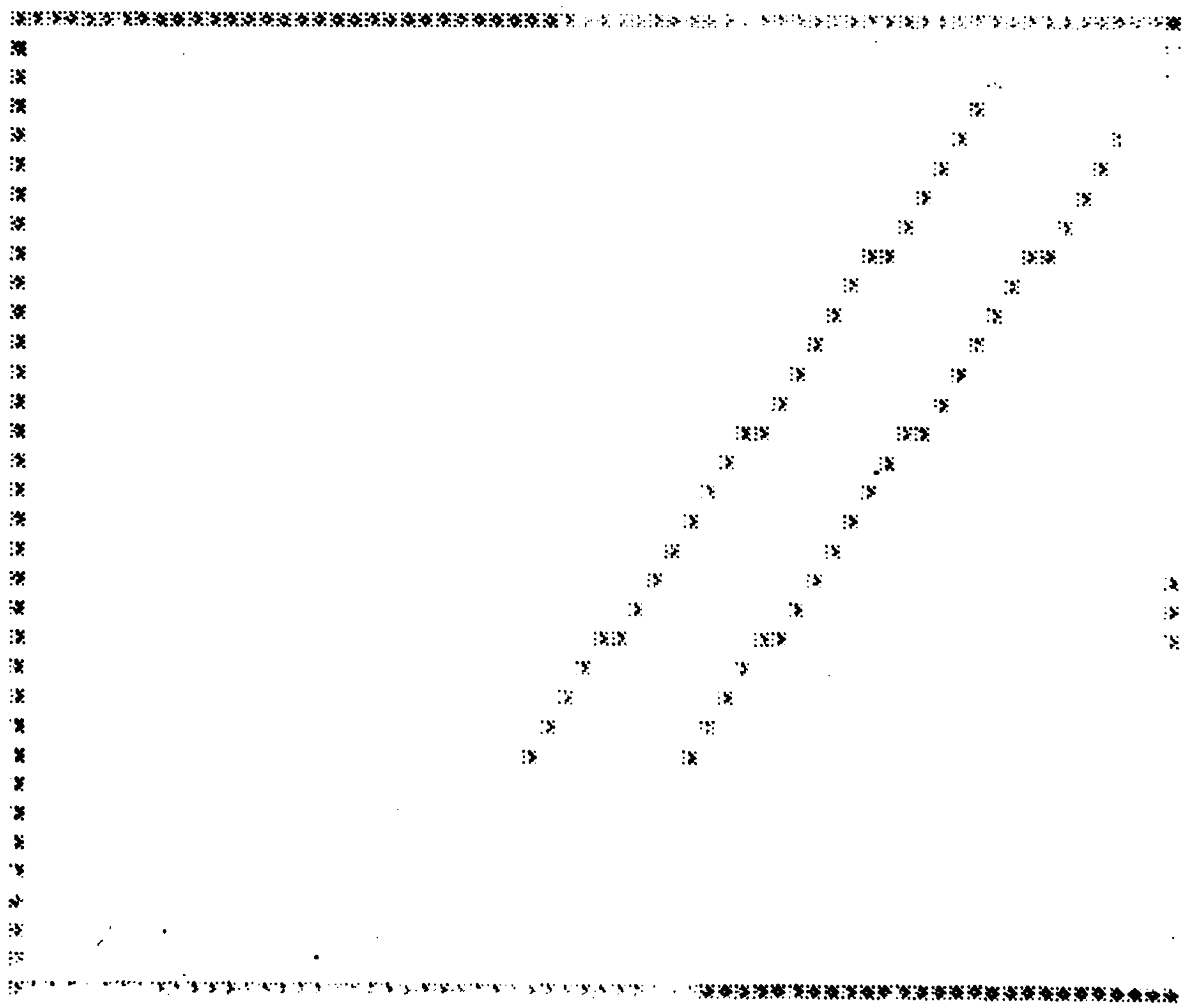
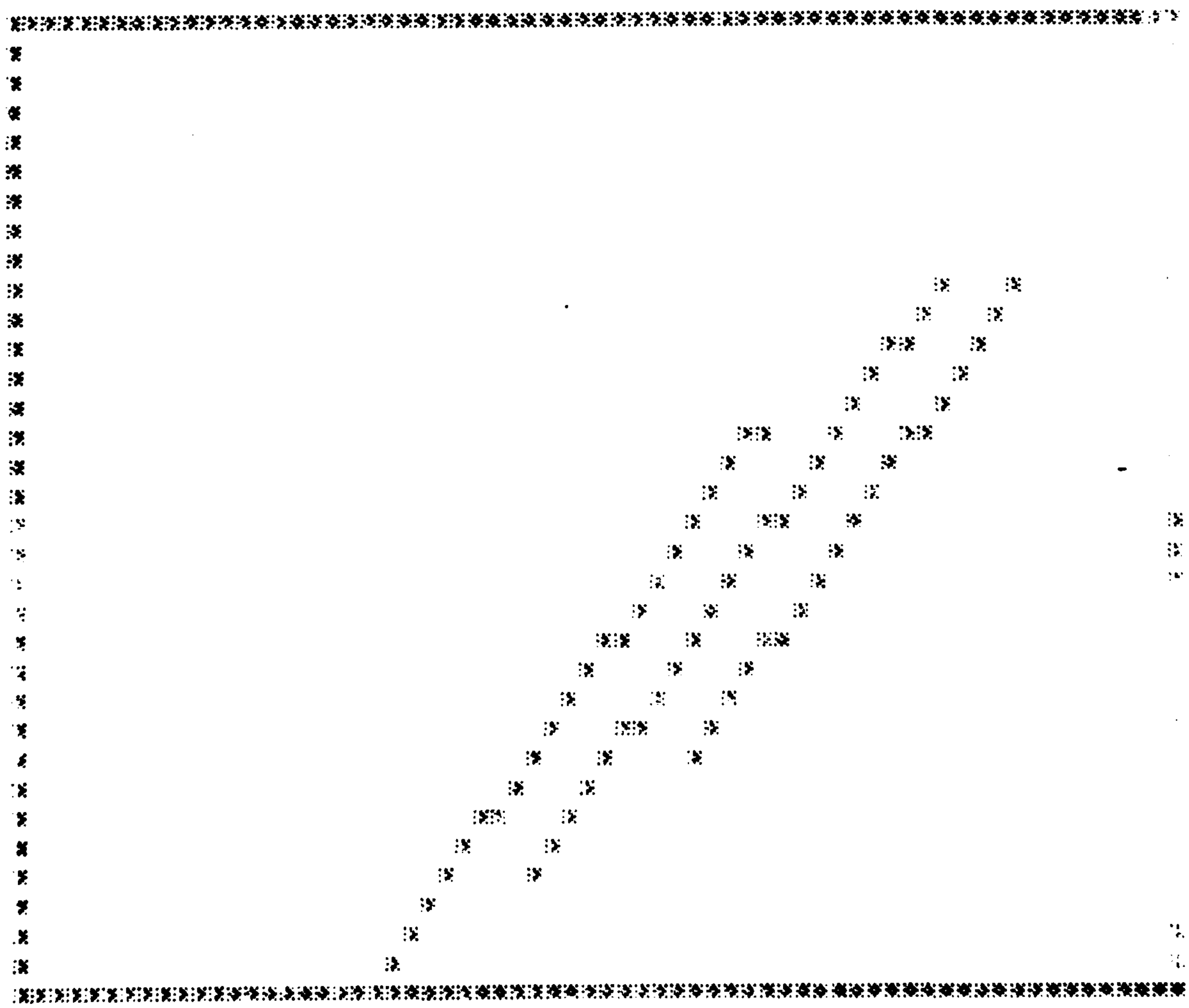


Figure 3.14. Example of the Hough and Inverse Hough Transforms with a Narrow Bridge. Hough (right) and inverse (below) based on upper part of Fig. 3.1. The boundaries and shadow border of the narrow bridge have been extracted.

Rho \ Theta	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	1	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	1	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	1	1	0	0	0	1	0	0	0	0	0	0
10	0	0	0	0	0	1	0	0	0	0	0	0	0
11	0	0	1	0	0	1	0	0	0	0	0	0	0
12	0	0	0	1	0	2	1	0	0	0	0	0	0
13	0	0	0	1	1	4	0	0	0	0	0	0	0
14	0	0	0	3	11	4	0	0	0	0	0	0	0
15	0	0	1	4	18	5	1	0	0	0	0	0	0
16	0	0	0	4	15	5	3	0	0	0	0	1	0
17	0	0	0	2	6	3	3	0	1	1	0	1	0
18	0	0	0	3	0	0	1	6	1	0	0	0	0
19	0	0	0	0	0	0	0	2	0	0	0	0	0
20	0	0	0	0	0	0	0	0	1	0	0	0	0
21	0	0	0	0	0	0	0	0	1	0	0	0	0
22	0	0	0	0	0	0	0	0	2	0	1	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0



have been isolated. In particular, one could do the following:

(1) decompose the image into textured segments (corresponding to, for example, water, land-urban, and land-rural via one of several proposed algorithms [17], [18], [37]. Then, long parallel lines over, say, water, are very probably bridges.

(2) perform a series of simple "environmental" tests on each candidate to determine its context. For example, one could determine a "global contrast", as a measure of the mean and variance of a block of pixels on either side of the lines. Those over water would tend to a uniformity and equality (i.e., similar means and small variances) for blocks or pixels on either side, since these would correspond to water. Another simpler procedure is to calculate the "edginess" of the region around a bridge candidate. Edginess is taken to be the number of detected edge points in the region divided by the total number of points in the region. (The edge detection algorithm of Appendices III-3 and III-5 determine this number automatically for the region contained within the subimage processed). Edginess is a common textural statistic and, again, we would expect the water regions (i.e., subimages containing bridges over water) would have a much lower amount of edginess than other regions, due to its high uniformity.

As an example, we shall use the edginess value defined above to attempt a classification of four potential bridges. As before, these candidates are taken to be sets of long, close parallel lines extracted from image segments. The four are those from Figures 3.13 and 3.14 (one each), plus those illustrated by the image and edge-detected images of Figures 3.16 and 3.17. The Hough transform matrices of these images are shown in Figure 3.15 with the bridge candidates circled. From experience, we may set the edginess threshold at, say, 0.20. Those candidates located in images with edginess values less than 0.20 will be judged as being over water, and therefore, bridges. Results are shown in Table 3.1.

Theta

									10	11	12	
1	0	0	0	0	0	0	0	0	0	0	0	0	
2	3	0	0	0	0	1	1	0	0	0	0	1	
3	1	0	0	0	1	0	1	0	0	1	0	3	
4	1	0	0	0	1	0	0	1	0	0	3	1	
5	10	1	0	0	0	1	1	0	0	2	1	2	
[illegible]	0	3	2	3	2	6	0	2	1	1	0	0	
[illegible]	3	12	4	4	4	4	0	0	0	0	0	0	
[illegible]	4	12	6	4	2	9	7	4	2	2	0	0	
[illegible]	3	3	0	0	2	4	4	2	1	0	5	1	
[illegible]	0	10	5	0	2	0	2	1	1	0	7	1	
[illegible]	0	2	3	2	0	2	0	1	5	0	0	3	
[illegible]	3	1	2	0	1	0	0	0	3	0	0	3	
[illegible]	2	0	3	0	1	1	0	1	2	0	0	1	
[illegible]	1	1	0	2	1	0	1	1	2	1	1	4	
[illegible]	7	3	0	2	0	0	0	0	0	1	0	1	
[illegible]	0	0	0	0	0	2	0	2	0	0	1	2	
[illegible]	1	1	2	0	1	1	0	0	0	0	1	2	
[illegible]	1	2	1	1	2	0	0	0	0	0	0	0	0
[illegible]	0	0	0	0	0	0	0	0	0	0	0	0	0
[illegible]	0	0	1	0	0	0	0	0	1	0	0	0	0
[illegible]	0	0	0	0	2	0	0	0	1	0	0	0	0
[illegible]	0	0	0	0	1	0	0	0	0	0	0	0	0
[illegible]	0	0	0	0	0	0	0	0	0	0	0	0	0
[illegible]	0	0	0	0	0	0	0	0	0	0	0	0	0

ANGLE DIVISION [illegible]

(a)

Theta

												12	13
1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	1	0	1	0	0	0	0	0
3	0	1	0	0	0	0	0	0	0	0	0	[illegible]	[illegible]
4	0	0	0	0	0	1	0	0	1	0	1		
5	0	0	2	2	2	0	0	0	0	1	0	1	
6	0	0	1	0	3	2	1	0	0	1	7	3	
7	0	1	0	1	1	0	0	1	0	0	4	3	
8	1	0	1	0	1	4	2	1	0	0	7	3	
9	2	1	1	2	2	0	2	3	1	1	7	2	
[illegible]	7	1	1	0	0	0	1	0	2	1	3		
[illegible]	15	1	4	0	0	0	0	0	1	0	3		
[illegible]	14	4	0	1	0	0	0	0	0	0	0		
[illegible]	12	1	0	0	0	0	0	0	0	1	1		
[illegible]	4	1	0	1	0	0	0	0	0	0	1		
[illegible]	0	1	0	0	0	0	0	0	0	0	0		
[illegible]	0	0	0	0	0	0	0	0	0	0	0		
[illegible]	0	0	0	0	0	0	0	1	0	0	0		
[illegible]	0	0	0	0	0	0	0	0	0	0	0		
[illegible]	0	0	0	0	0	0	0	1	0	0	0		
[illegible]	0	0	0	0	0	0	0	0	0	0	0		
21	0	0	0	0	0	0	0	0	0	0	0		
22	0	0	0	0	0	0	0	0	0	0	0	1	
23	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	1	1	
24	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0	0	0

ANGLE DI[illegible]

(b)

Figure 3.15 Hough Transform Matrices

(a) For Fig. 3.16.
(b) For Fig. 3.17.

The potential bridges (long, close, parallel lines) are circled.

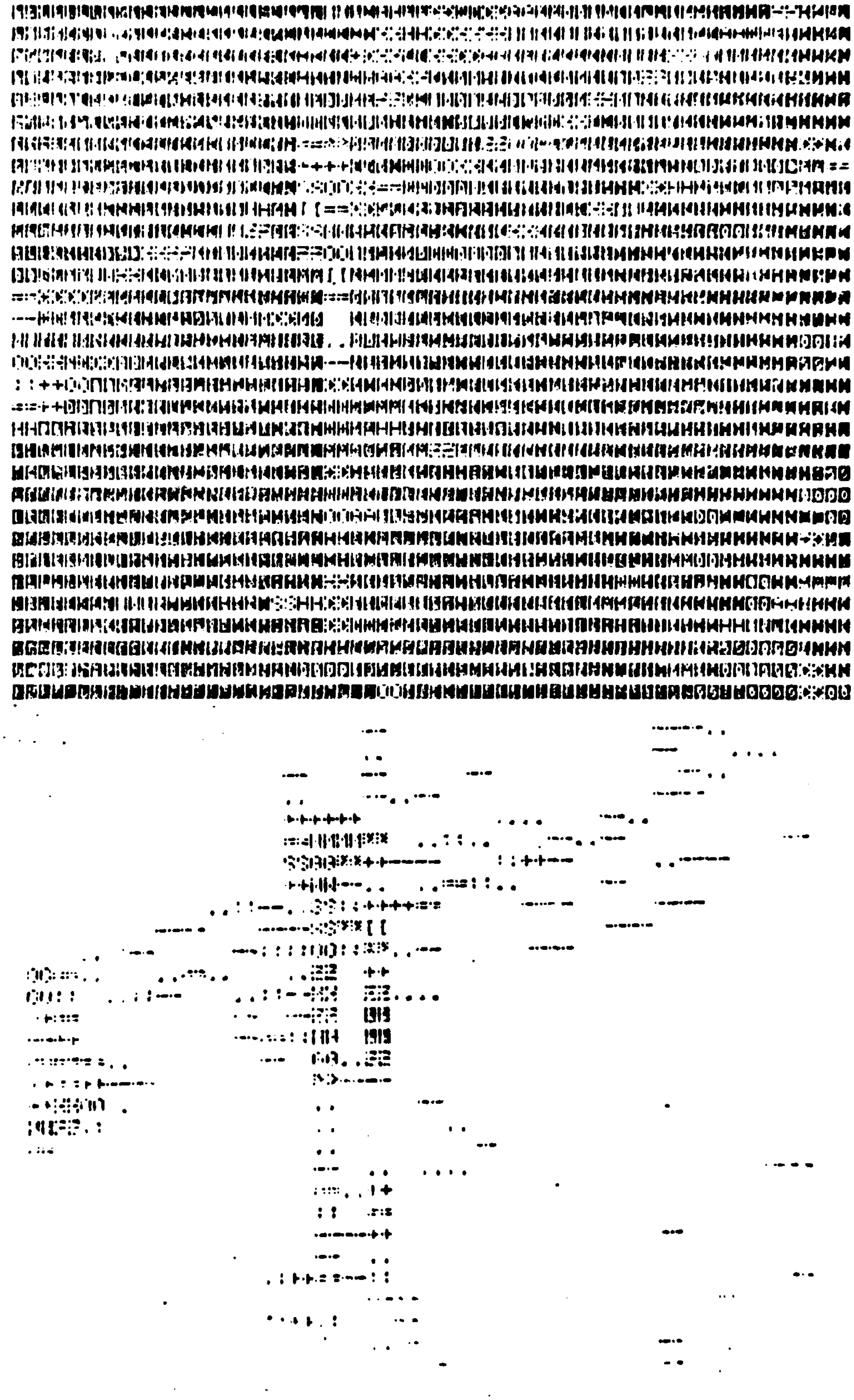


Figure 3.16. Raw Image and Edge-Detected Version: Highway and Overpass

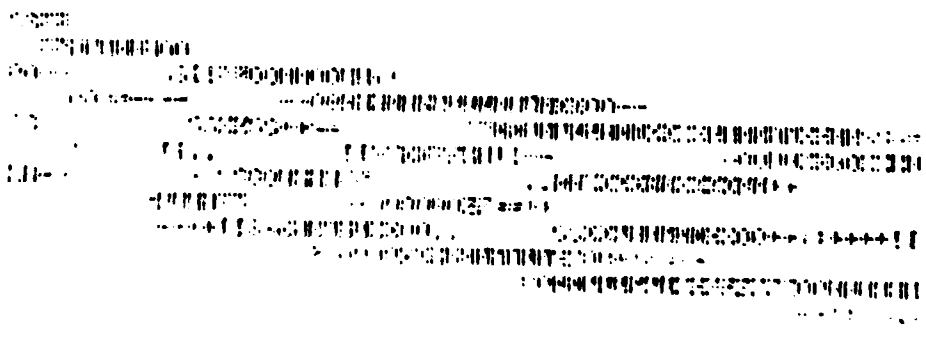


Figure 3.17. Raw Image and Edge-Detected Version: Bridge Over River

TABLE 3.1

Edginess of Bridge Candidates

Candidate from:	# edge points (excluding) bridge pts)	Edginess	Thres- hold	Classifi- cation	Actual
Figure 3.13	406	0.41	0.20	Non-bridge	Highway
Figure 3.14	84	0.09	0.20	Bridge	Bridge
Figure 3.16	313	0.31	0.20	Non-bridge	Highway/ Overpass
Figure 3.17	131	0.15	0.20	Bridge	Bridge

Summary

Bridge candidates are extracted from raw digitized images by performing a Hough transform on an edge detected image and keeping only those entries corresponding to long parallel lines that are close together. These are represented by matrix entries in the same column (same angle) and in rows that are adjacent or nearly adjacent. Further testing is then done on this small set of potential bridges to determine whether or not a bridge is present. An example using image edginess in the neighborhood of the candidate is given.

CHAPTER IV

THE DISCRETE MEDIAL AXIS TRANSFORM

Introduction

The Medial Axis Transform (MAT) or "prairie fire" skeleton of an image was first defined by Blum [1] in 1964. There was a flurry of activity, both theoretical and practical, for the next 6 years, ending about 1970, (see refs. [3]-[8], [19], [20], [22]-[26], [29], [32], [34]). Since 1970, the MAT has been mentioned in some texts ([9], [11], [30], [36]), and some investigators have used it with success in certain problems ([28], [38]). Part of the problem with the MAT which caused a decline of interest in 1970 was that the existing algorithms were either difficult to understand, or they ran too long on the computers then available. Moreover, the results were highly dependent upon the orientation of the object being transformed; that is, the algorithms were very sensitive to rotation in the plane. More recently, Wall and his associates [38] have employed an algorithm essentially identical to the one described later in this paper, and Pavlidis [28] has used the MAT as the basis for defining a variety of shape descriptors for use in pattern classification schemes.

The present study was provoked by a need to find feature extraction techniques which could be used to identify bridges in

aerial photographs. As noted earlier, one of the features of a bridge in a photograph that is potentially useful as a discriminator in a classification scheme is the aspect ratio. Another useful feature is the length of the bridge. Virtually all bridges are long, and have aspect ratios which are high. Thus, a test of these two features can eliminate many objects from further consideration in an automatic bridge identification scheme.

Both the aspect ratio and the length of an object are easily extracted from the object's MAT by simple syntax tests. The definitions of a grammar and the test itself are not discussed here. Instead, the computation of the MAT in a binary image by thinning is considered, and extensions to grayscale images are examined.

A Binary Thinning Algorithm

The "prairie fire" definition of the MAT can be used to generate an algorithm for the computation of the MAT. For example the white areas of a binary image can be skeletonized or thinned quite easily by application of the thinning algorithm described below. Black areas can be thinned by complementing the image both before and after application of the thinning algorithm.

This thinning algorithm operates by using a 3 x 3 pixel mask defined as

```
A B C
D E F
G H I
```

where the candidate for replacement is the element lying under E.

The original image is copied, and then the mask is moved over the original image of size m x n such that the mask does not extend past the edges of the original mask. Thus, the thinning algorithm does not have any effect on the border pixels in the original image.

Initialization is effected by setting a flag and defining a quantity called "pass number" and setting it equal to 1. Then the mask is slid across the image beginning with element (2,2) under E and ending with element (m-1,n-1) lying under E. At each point, a test is made to see if the element lying under E is a 1. If not, the mask is slid to the next position. If the element under E was a 1, then a test is made of the set of elements lying under mask elements B, D, F, and H. If two or three of these elements are 1's, then the path between each pair of 1's is examined to see if they are connected by 1's where element E is not an allowed path member. For each pair there are 2 paths around the perimeter of the mask. In order to pass the test of connectivity between 1's, only one of these paths need be all 1's. If the test is passed by each pair of 1's, then the element lying under E in the original image is replaced with a 0 in the copied image and the flag is reset. If there were not two or three 1's in the set B, D, F, H, or if the connectivity test is failed by any pair of 1's, then the mask is slid to the next position with no action taken on element E. At the conclusion of each pass over the image, the flag is tested. If it is reset, the pass number is incremented, the original image is replaced with the copied image, the flag

is again set, and the algorithm proceeds anew. If the flag has not been reset, then no elements were replaced in the preceding pass, and the algorithm halts by outputting the copied image.

The operation of the algorithm is shown by Figures 4.1, a, b, and c.

Computation of MAT by Thinning

The conversion of the binary thinning algorithm just described to the computation of the medial axis transform is quite easy. The only additional work required is bookkeeping: i.e., a list must be maintained whose elements are certain of the replaced elements in the image being thinned. The criterion for inclusion in this list is that the element being replaced have two members of its set of neighbors, B, D, F, and H equal to zero. This is the defining relation for a skeleton element.

The actual list consists of more than just elements. It has four fields for each list member. The first two fields are the coordinates of the replaced element. The next field is the pass number in which the replacement was effected. The fourth field is of variable length and consists of a coded representation of where the neighbors of the replaced elements were 1's in the set B, D, F, and H. The reason that this field is of variable length is that the MAT of an object computed by this algorithm consists of all the replaced elements in union with the elements remaining as 1's in the thinned image. These elements are assigned pass number 0 in the list, and they have no neighbors when replaced since they were never replaced.

0	0	0	0	0	0	0	0
0	1	1	1	1	0	0	0
0	1	1	1	1	0	0	0
0	1	1	1	1	1	0	0
0	1	1	0	1	1	0	0
0	0	0	0	1	1	1	0
0	0	0	0	1	1	0	0
0	0	0	0	0	0	0	0

(a)

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	0
0	0	1	0	1	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(b)

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0
0	0	1	0	1	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(c)

0	0	0	0	0	0	0	0
0	1	0	0	1	0	0	0
0	0	1	1	0	0	0	0
0	0	1	0	1	1	0	0
0	1	1	0	0	0	0	0
0	0	0	0	0	1	1	0
0	0	0	0	1	1	0	0
0	0	0	0	0	0	0	0

(d)

Figure 4.1. Test Image; a) Original Image; b) After First Pass of Thinning Algorithm; c) Thinned Image After Third Pass; d) Skeleton of Image from MAT Algorithm.

The coding for the sequence is based upon the well known chain code mask [30].

7	0	1
6		2
5	4	3

As an example of the application of this algorithm, consider the image shown in Figure 4.1a. The skeleton of this image is shown in Figure 4.1d and its MAT in list form is shown below:

2	2	1	24
2	5	1	46
4	6	1	46
5	2	1	02
5	3	1	06
7	5	1	02
7	6	1	06
3	3	2	24
3	4	0	--
4	3	0	--
4	5	0	--
6	6	0	--
6	7	0	--

The basis of the algorithm is the prairie fire concept. A fire lit simultaneously at all points on the perimeter of an area will burn towards the interior from the perimeter as a wave would propagate except that there is no superposition. The points where the propagating fire fronts meet are called quench points and the distance to the perimeter from each

quench point is called the quench distance. The set of all quench points is the skeleton of the object and the set of all quench points and quench distances is the medial axis transform of the object.

In the discrete plane, the prairie fire concept can be implemented by thinning the points on the edge of an object and keeping track of those points which have two edges, i.e., corner points, as the object is repeatedly thinned. Keeping track of the orientation of the corners allows reconstruction.

In the continuous plane, the MAT is s set of connected line segments made up of straight lines and arcs of parabolas [25]. In the discrete plane, the line segments are not necessarily connected as the example in Figure 4.1 shows, (see also [9], page 331). Moreover, there is some visible distortion of the MAT as shown in Figure 4.2b which is a function of the scan directions as the mask is passed over the image. This factor must be considered in any grammar which is defined to permit syntax testing of the skeleton or of the MAT itself.

The MAT in the discrete plane is also somewhat sensitive to rotation, as is discussed in [9]. This sensitivity is not significant if the definition of the grammar takes it into account. Hence, it will not be further considered in this paper inasmuch as it will be dealt with in a future paper on grammars for syntax testing of MAT's.

Montanari's Method

Montanari [24] formulates the determination of the discrete MAT as an optimal policy problem, that is, solving the

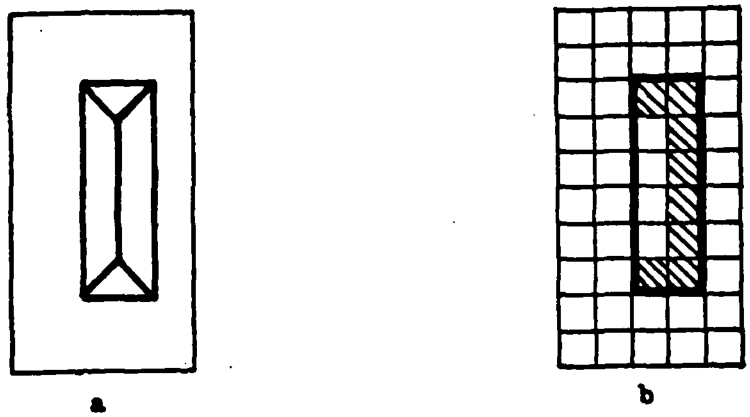


Figure 4.2. Rectangles and Their MAT's: a) continuous; b) discrete.

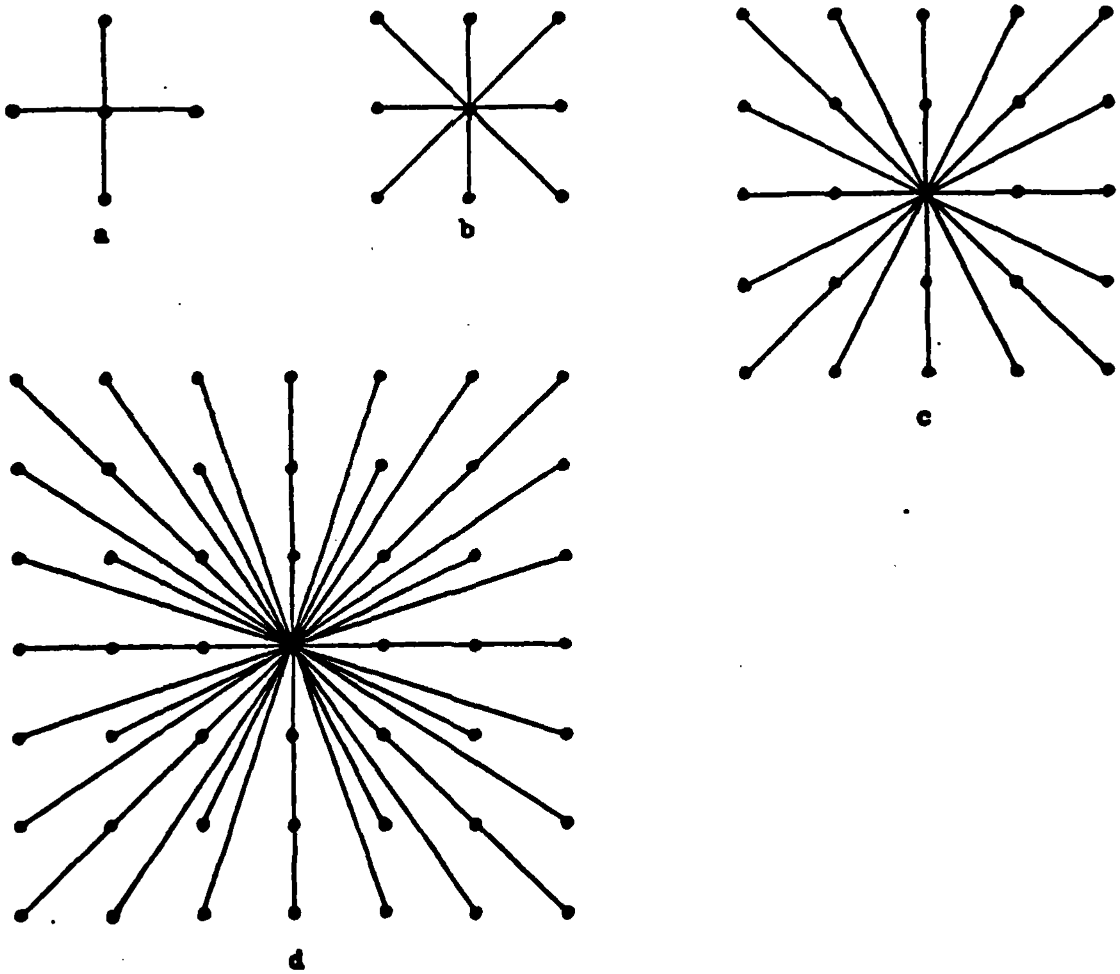


Figure 4.3. Recticular Networks: a) $n=0$; b) $n=1$; c) $n=2$; d) $n=3$.

problem of finding the shortest path through a network. For the discussion that follows, one must realize that Montanari computes external MAT's, that is, MAT's which lie outside the boundaries of the objects. To arrive at MAT's which have a finite set of points on the skeleton, i.e., bounded MAT's, Montanari employs "hollow" objects for his examples, that is, he computes the MAT for either unbounded, simply connected objects, or for multiply connected objects and the MAT lies in the bounded exterior of the object.

He begins by defining a reticular network of order n as one where the slopes of the lines connecting the center of the network to the vertices in the first octant form a Farey sequence of order n (defined as the ordered sequence of all rational numbers between 0 and 1 with denominators less than or equal to n) as shown in Figure 4.3.

He then defines three sets. Set $\underline{U}$ is the set of all the vertices which are in one-to-one correspondence with the coordinates of the pixels in the image, i.e.,

$$\underline{U} = [\underline{P}' \mid 1 \le x_p \le r;\ 1 \le y_p \le s;\ (x_p, y_p) = \text{pixel coordinates}].$$

set $\underline{I}$ is the set of vertices defining the object:

$$I = [P' \mid a(P) = .\text{True}.]$$

i.e., the set corresponding to the pixels belonging to the object. Set $\underline{E}$ is the complement of the set $\underline{I}$, i.e., $\underline{E} = \underline{U} - \underline{I}$. Also defined is an extra vertex, P_N', which is assumed to be connected to every vertex $\underline{I}$ by an arc of length zero.

Now for every $P_i \varepsilon \underline{U}$, there is a minimum arc length connecting p_i' to P_N' which lies on a set of network rays. This minimum distance Montanari calls the quasi-Euclidean distance since from any n, n finite, $D_q(P_1,P_2) < D_E(P_1,P_2)$ and in the limit as $n \rightarrow \infty$, $D_q \rightarrow D_E$ where Dq is the minimum distance along the network paths and D_E is the usual Euclidean distance.

$$D_E = [(x_1 - x_2)^2 + (y_1 - y_2)^2]^{1/2} \tag{1}$$

A point P_i can be defined to be a skeleton point if and only if $P_i' \varepsilon \underline{E}$ and it does not lie on a minimal path from any other vertex P_j' to P_N'. Thus, for every vertex P_i', the minimum path to P_N' is found and all the P_j', $j \neq i$, lying on this path are removed from further consideration since they cannot be skeleton points. The length of the path is also found and is associated with P_i' as a function $T(P_i')$. The set, $\underline{S}$, of all P_i' which remains after the entire set $\underline{U}$ has been examined is the skeleton and the set of all the $T(P_i')$ defines the quench function.

Montanari determines the difference between the quasi-Euclidean distance and the true Eucludean distance by computing the error, e, where

$$e = \frac{T(P_1', P_2') - D_E(P_1', P_2')}{T(P_1', P_2')}$$

where P_1' and P_2' are any two points in the plane.

Montanari's Algorithms

Montanari has devised two algorithms to compute the discrete MAT. His first approach is to solve the set of equations

$$T_i = \min(t_{ij} + T_j) \quad i = 1,2,\ldots,N-1;\ j = 1,2,\ldots,N,\ j \neq i \quad (3a)$$

$$T_N = 0 \quad (3b)$$

where if the arc connecting vertices P_i' and P_j' exists, t_{ij} is the length of this arc; otherwise $t_{ij} = \infty$, and where i is an index indicating the forward raster sequence number. The initial condition is that $T_i = t_{iN}$. Then the iterative formula is

$$T_i^k = \min\,(t_{ij} + T_j^k,\ t_{ir} + T_r^{k-1}) \qquad i = 1,2,\ldots,N-1 \quad (4)$$

$$P_j' \in \underline{E}_{i1}^k$$

$$P_r' \in \underline{E}_{i2}^k$$

where $\underline{E}_{i1}^k$ is the set of vertices P_j' for which the value T_j^k was computed in the k^{th} iteration, and $\underline{E}_{i2}^k = \underline{U} - \underline{E}_{i1}^k[P_i]$. The iteration stops when $T_i^{m+1} - T_i^m = T_i$ for all P_i.

Now the order in which the T_i^k are computed is arbitrary. However, the rate of convergence is a function of the order of computation. Montanari has found an order of computation for which convergence is achieved in only two iterations. The secret to this rapid convergence is the fact that the image is a rectangular array and that information gained during the first iteration is used to reduce the computation necessary during the second iteration.

The way this works is that by scanning the array in a forward raster sequence in the first iteration and with a backward raster sequence in the second iteration, $T_j^2 = T_j$ for all P_j' in two iterations. Moreover, he only considers $P_i' \varepsilon \underline{E}$ since for $P_i \varepsilon \underline{I}$, $T_i = 0$, and in the first iteration, he can ignore the vertices $P_r' \varepsilon \underline{E}_{i2}$ because $T_r^0 = \infty$. Other shortcuts are based upon the fact that the set E_{11}^1 is empty, hence $T_1^1 = \infty$. and that $\underline{E}_{12}^2 = \underline{E}_{i1}^1 = \underline{E}_{i1}$ and that $\underline{E}_{i2}^1 = \underline{E}_{i1}^2 = E_{i2}$ so that direct use of T_i^1 can be made in the second iteration.

The algorithm is as follows:

a) Let $T_i^1 = 0 \quad \text{if } P_i \ \varepsilon \ \underline{I}$ (5a)

$$T_1^1 = \infty \quad \text{if } P_1^1 \ \varepsilon \ \underline{E} \tag{5b}$$

$$T_i^1 = \min_{P_j' \varepsilon \underline{E}_{i1}} (t_{ij} + T_j^1) \quad \text{if } P_i \ \varepsilon \ \underline{E} \tag{5c}$$

where $\underline{E}_{i1} = [P_j' \ j < i]$

b) Let

$$T_1^2 = \begin{cases} (1) \min\limits_{P_j' \varepsilon \underline{E}_{i2}} (t_{ij} + T_j^2, T_i^1) \quad i = n-1, \ldots .1 & \text{if } P_i' \ \varepsilon \ \underline{E}, \text{ and} \\ (2) \quad 0 & \text{if } P_i' \ \varepsilon \ \underline{I}. \end{cases}$$

where $\underline{E}_{i2} = [P_j' \mid j > i]$, and $\underline{U} = \underline{E} \vee \underline{I} = \underline{E}_{i1} \vee \underline{E}_{i2} \vee [P_i']$.

c) Let $\underline{S} = [P_k \mid P_k' \ \varepsilon \ \underline{E}$ for every P_i' directly connected to P_k', $T_k^2 \neq T_i^2 - t_{ik}]$ (7)

d) Associate to every $P_k \ \varepsilon \ \underline{S}$ the parameter T_k.

As an example of Montanari's first algorithm, consider the example of Figure 4.1a. We first complement this image to yield the image shown in Figure 4.4a. Then we "countersign" all the points which are in the set $\underline{I}$ with 0's and leave the remaining points blank, as shown in Figure 4.4b. Then, employing the reticular grid of order n = 2, we have after the first iteration the representation shown in Figure 4.4c. The second iteration produces the representation shown in Figure 4.4d. We then examine all the $P_k' \varepsilon \underline{E}$ for the test specified in (7), and assign to each survivor P_k' its associated value T_k. The resulting MAT is shown in Figure 4.4e. Note that this MAT is fuller, i.e., has more points on the skeleton than does the MAT in Figure 4.1d. Also different are the distances or quench function. This is to be expected since the local vicinity, i.e., mask, in the algorithm used to compute the MAT in Figure 4.4e is larger than that used to compute the MAT of Figure 4.1d. In order to make a more realistic comparison between these two algorithms, the first Montanari algorithm was reapplied with n = 0. The results are shown in Figure 4.4f.

We note that this MAT, while more similar to the MAT of Figure 4.1d than the one computed with n = 2, is still different. Thus, the choice of algorithm is another factor which must be considered in the design of a grammar for describing the syntax of the MAT.

Montanari's second algorithm is a Dantzig (one-pass) algorithm. The algorithm proceeds as follows:

1	1	1	1	1	1	1	1
1	0	0	0	0	1	1	1
1	0	0	0	0	1	1	1
1	0	0	0	0	0	1	1
1	0	0	1	0	0	1	1
1	1	1	1	0	0	0	1
1	1	1	1	0	0	1	1
1	1	1	1	1	1	1	1

a

0	0	0	0	0	0	0	0
0					0	0	0
0					0	0	0
0						0	0
0			0			0	0
0	0	0	0				0
0	0	0	0			0	0
0	0	0	0	0	0	0	0

b

0	0	0	0	0	0	0	0
0	1	1	1	1	0	0	0
0	1	2	2	$\sqrt{2}$	0	0	0
0	1	2	$\sqrt{5}$	$\sqrt{2}$	1	0	0
0	1	2	0	1	$\sqrt{2}$	0	0
0	0	0	0	1	$\sqrt{2}$	1	0
0	0	0	0	1	2	0	0
0	0	0	0	0	0	0	0

c

0	0	0	0	0	0	0	0
0	1	1	1	1	0	0	0
0	1	2	2	1	0	0	0
0	1	$\sqrt{2}$	1	$\sqrt{2}$	1	0	0
0	1	1	0	1	1	0	0
0	0	0	0	1	$\sqrt{2}$	1	0
0	0	0	0	1	1	0	0
0	0	0	0	0	0	0	0

d

•	•	•	•	•	•	•	•
•	1	•	•	1	•	•	•
•	•	2	2	•	•	•	•
•	1	$\sqrt{2}$	•	$\sqrt{2}$	1	•	•
•	1	1	•	1	1	•	•
•	•	•	•	1	$\sqrt{2}$	1	•
•	•	•	•	1	1	•	•
•	•	•	•	•	•	•	•

e

•	•	•	•	•	•	•	•
•	1	•	•	1	•	•	•
•	•	2	2	•	•	•	•
•	•	2	•	2	•	•	•
•	1	•	•	•	•	•	•
•	•	•	•	•	2	•	•
•	•	•	•	1	•	•	•
•	•	•	•	•	•	•	•

f

Figure 4.4. Montanari's First Algorithm: a) Original Image; b) Countersigned Image; c) After First Iteration, n=2, d) After Second Iteration, n=2; e) MAT for n=2; f) MAT for n=0.

(1) Examine all P_k' to find t_{kN} such that $t_{kN} = \min t_{jN}$. Obviously $T_N = 0$, and $T_k = t_{kN}$. Then define the sets $\underline{E}_1^2 = [P_N]$, $\underline{E}_2^2 = \underline{U} - \underline{E}_1^2$.

(2) Compute the values

$$T_h^m = \min_{P_x' \in \underline{E}_1^m} (t_{hx} + T_x) \quad \forall P_h' \in \underline{E}_2^m \tag{8}$$

(3) Compute

$$\underline{E}_1^{m+1} = \underline{E}_1^m = [P_k'] \tag{9a}$$

$$\underline{E}_2^{m+1} = \underline{E}_2^m - [P_k'] \tag{9b}$$

If E_2^{r+1} is empty, stop; otherwise repeat steps (2) and (3). For the vertices P_k accepted in the m^{th} step, remember the vertex (or vertices) P_x' for which $(t_{kx} + T_x)$ is the least in (8), i.e., for which $T_x = T_h - t_{hx}$, and countersign all the P_x' that are encountered in this manner. The non-countersigned vertices belong to the set $\underline{S}$.

The algorithm is demonstrated for n = 0 by again considering the image in Figure 4.1a. The intermediate operations and the results are shown in Figure 4.5. We note that Figure 4.5c is identical to Figure 4.4f, demonstrating the equivalence of these two algorithms.

Montanari also proposes a method of eliminating insignificant skeleton points by introducing a threshold function into his algorithms. For the first algorithm, this involves changing the criterion for inclusion in $\underline{S}$ as follows;

•	•	•	•	•	•	•	•
•					•	•	•
•					•	•	•
•						•	•
•			•			•	•
•	•	•	•				•
•	•	•	•			•	•
•	•	•	•	•	•	•	•

a

•	•	•	•	•	•	•	•
•	1	$\underline{1}$	$\underline{1}$	1	•	•	•
•	$\underline{1}$	2	2	$\underline{1}$	•	•	•
•	$\underline{1}$	2	$\underline{1}$	2	$\underline{1}$	•	•
•	1	$\underline{1}$	•	$\underline{1}$	$\underline{1}$	•	•
•	•	•	•	$\underline{1}$	2	$\underline{1}$	•
•	•	•	•	1	$\underline{1}$	•	•
•	•	•	•	•	•	•	•

b

•	•	•	•	•	•	•	•
•	1	•	•	1	•	•	•
•	•	2	2	•	•	•	•
•	•	2	•	2	•	•	•
•	1	•	•	•	•	•	•
•	•	•	•	•	2	•	•
•	•	•	•	1	•	•	•
•	•	•	•	•	•	•	•

c

Figure 4.5. Montanari's Second Algorithm for n=0; a) Original Image; b) After Application of Algorithm; c) MAT Resulting from Algorithm.

$$\underline{S} = [P_i \mid P_i' \in \underline{E}; \text{ for every } P_j',\ T_j - T_i < K \cdot t_{ij}] \quad (10)$$

and for the second algorithm, countersigning the vertices for which $T_x \leq T_h - k \cdot t_{hk}$, where K is the threshold coefficient, $0 \leq K \leq 1$.

The major use of this threshold function is to eliminate skeleton noise introduced when $n > 0$. Montanari suggests as a rule of thumb that $\underline{K} \leq 0.70$, and in his paper [24] he gives an example for three values of the threshold showing 0.70 to yield the best results.

Montanari's Gray Weighted Skeleton

Montanari and Levi [23] are able to extend Montanari's algorithms for finding MAT's of binary images to gray scale images by defining a new metric for measuring distances in the reticular networks. They do this by first specifying some gray function f(x,y) in the real plane which interpolates the gray values found in the discrete image. The metric $t_{ij,rs}$ is defined as

$$t_{ij,rs} = \int_l f \, dl \quad (11)$$

where the integration path lies between P(i,j) and P(r,s).

In the case of binary images, the first of Montanari's algorithms converges in two iterations. Now, however, the first method cannot be assumed to converge in two iterations. The algorithm is modified in the following manner. We now assume that the skeleton is internal to the object instead of external to it as before. Hence, we place zeroes at all the vertices external to the image and blanks at the remaining vertices.

Then each element is considered sequentially and a new value is computed from the following formula:

$$b_{ij} = \min_{[P_{rs}']} (b_{ij}, b_{rs} + t_{ij,rs})$$

where $[P_{rs}']$ is the set of vertices directly connected to P_{ij} for which new b_{rs} have already been computed, as before. The method is applied with forward raster sequence alternating with backward raster sequence until the image is unchanged in both directions.

The computation of $t_{ij,rs}$ can involve considerable work depending upon the interpolating function chosen. For the step function, the work is easy since

$$t_{ij,rs} = (1/2)(a_{ij} + a_{rs}) \cdot t_{ij',rs} \qquad (13)$$

where $t_{ij,rs}$ is the true length of the arc connecting P_{ij}' and P_{rs}'.

The skeleton points are those for which

$$b_{ij} > \max_{[P_{rs}]} (0, b_{rs} - t_{ij,rs}) \qquad (14)$$

where $[P_{rs}]$ is the set of all the vertices directly connected to P_{ij}.

Figures 4.6, 4.7, and 4.8, taken from [23], show the method applied to a digitized image of a human chromosome.

Montanari and Levi do not explicitly describe the alterations necessary to the second algorithm in order to compute

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 2 2 2 1 1 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 3 6 6 6 5 3 1 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 2 6 6 6 6 6 6 4 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 3 6 6 6 6 6 6 6 4 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 3 6 6 6 6 6 6 6 6 3 1 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 2 4 6 6 6 6 6 6 6 6 4 2 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 2 3 4 6 6 6 6 6 6 6 6 3 1 1 1 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 3 5 6 6 6 6 6 6 6 6 4 3 2 1 1 3 3 2 1 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 2 3 6 6 6 6 6 6 6 6 4 2 2 1 1 1 2 5 6 6 3 1 0 0
0 0 0 0 0 0 0 0 0 0 1 2 3 4 5 5 6 6 6 6 6 6 6 4 3 1 1 1 1 1 2 4 6 6 6 6 3 1 0
0 0 0 0 0 0 0 0 0 1 2 4 6 6 6 6 6 6 6 6 6 4 3 1 1 1 1 1 1 2 4 6 6 6 6 6 5 2 0
0 0 0 0 0 0 0 0 2 4 6 6 6 6 6 6 6 6 5 4 3 2 1 1 1 1 1 2 4 5 6 6 6 6 6 6 4 2 0
0 0 0 0 0 0 0 1 4 6 6 6 6 6 6 6 5 3 2 1 1 1 1 1 3 3 5 6 6 6 6 6 6 6 6 6 3 1 0
0 0 0 0 0 0 1 3 6 6 6 6 6 6 6 4 2 1 1 1 1 1 3 4 6 6 6 6 6 6 6 6 6 6 6 3 2 0 0
0 0 0 0 0 1 3 6 6 6 6 6 6 4 3 1 1 1 1 2 3 4 6 6 6 6 6 6 6 6 6 6 6 5 3 1 0 0 0
0 0 0 0 0 1 5 6 6 6 6 6 4 2 1 1 1 1 3 5 6 6 6 6 6 6 6 6 5 5 5 4 3 2 1 0 0 0 0
0 0 0 0 0 2 6 6 6 6 6 6 2 1 1 1 2 3 6 6 6 6 6 6 6 5 3 3 2 1 1 1 1 0 0 0 0 0 0
0 0 0 0 1 4 6 6 6 6 6 3 1 1 1 3 4 6 6 6 6 6 6 5 3 1 1 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 1 5 6 6 6 6 4 1 1 1 3 6 6 6 6 6 6 6 4 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 2 5 6 6 6 4 2 1 1 3 5 6 6 6 6 6 5 3 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 4 6 6 6 4 3 2 2 3 6 6 6 6 6 4 2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 3 5 6 6 5 3 3 3 5 6 6 6 6 4 3 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 4 6 6 6 4 4 6 6 6 6 6 6 4 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 3 5 6 6 6 6 6 6 6 6 6 6 5 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 2 5 6 6 6 6 6 6 6 6 6 4 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 2 6 6 6 6 6 6 6 6 6 3 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 2 5 6 6 6 6 6 6 6 3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 3 6 6 6 6 6 6 6 4 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 3 6 6 6 6 6 6 6 4 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 3 5 6 6 6 6 6 6 6 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 2 3 5 5 6 6 6 6 6 4 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 2 4 4 3 4 6 6 6 6 6 4 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 3 4 5 4 3 5 6 6 6 6 6 3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 1 3 5 5 3 2 3 5 6 6 6 6 6 4 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 3 4 6 4 3 2 2 4 6 6 6 6 6 5 3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 2 5 6 6 3 2 1 2 4 6 6 6 6 6 6 5 3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 2 5 6 6 5 3 1 1 3 6 6 6 6 6 6 6 6 3 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 2 4 6 6 6 6 2 1 2 3 6 6 6 6 6 6 6 6 5 3 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 1 3 6 6 6 6 5 2 1 1 2 5 6 6 6 6 6 6 6 6 4 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 3 6 6 6 6 6 4 2 1 1 2 4 6 6 6 6 6 6 6 6 3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 2 5 6 6 6 6 6 3 1 1 1 2 3 5 6 6 6 6 6 6 4 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 3 6 6 6 6 6 5 2 1 1 1 1 2 5 6 6 6 6 6 4 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 2 5 6 6 6 6 6 5 3 1 1 1 1 1 3 4 6 6 4 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 2 6 6 6 6 6 6 6 4 2 1 1 1 1 1 2 2 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 3 6 6 6 6 6 6 6 5 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 1 3 5 6 6 6 6 6 6 6 3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 2 3 6 6 6 6 6 6 6 3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 3 6 6 6 6 6 6 4 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 1 2 4 6 6 6 6 3 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 1 2 3 3 3 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

Figure 4.6. Human Chromosome (from [23]).

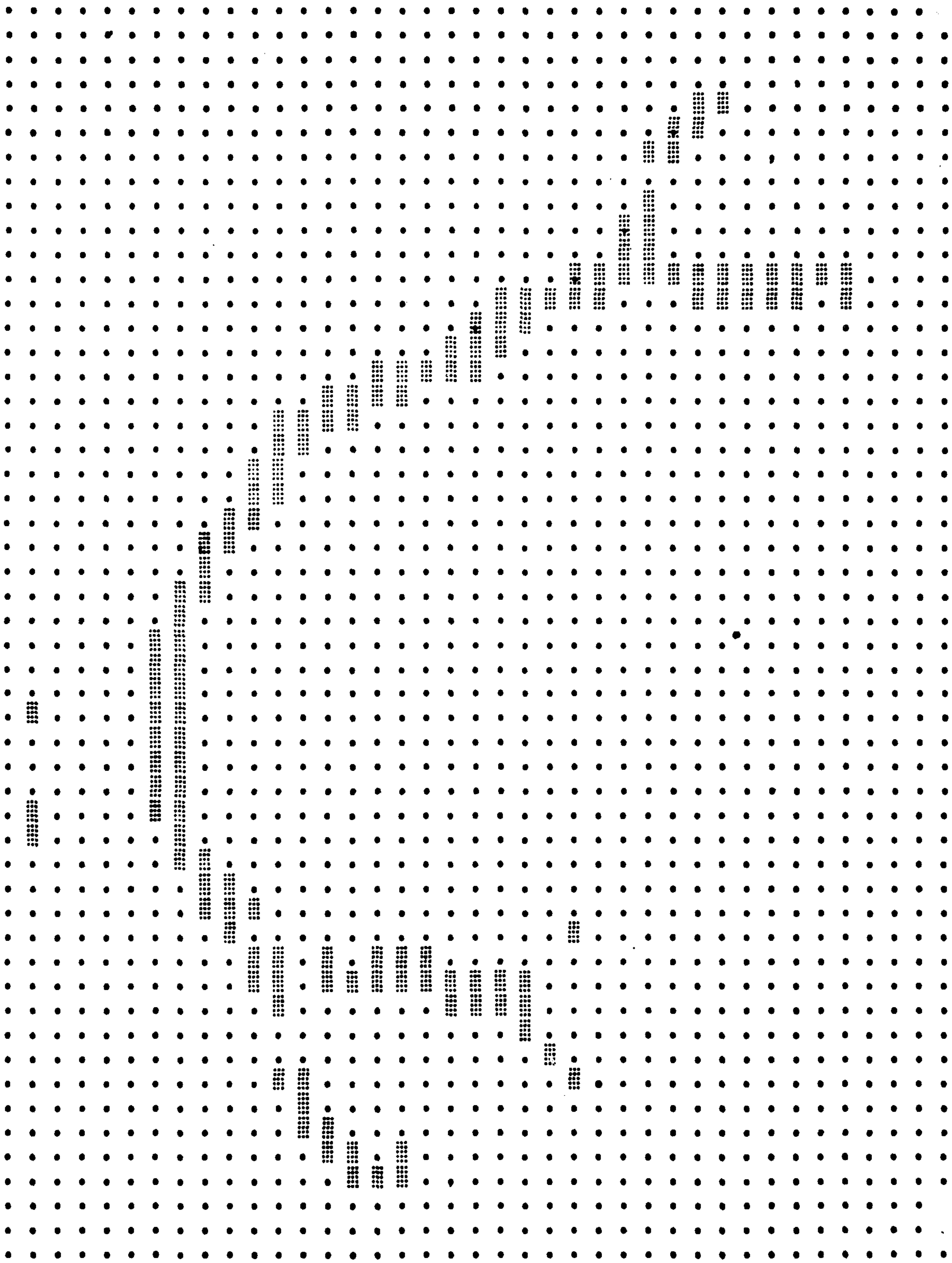


Figure 4.7. Black and White Skeleton of Human Chromosome; n=2; K=0.80 (from [23].

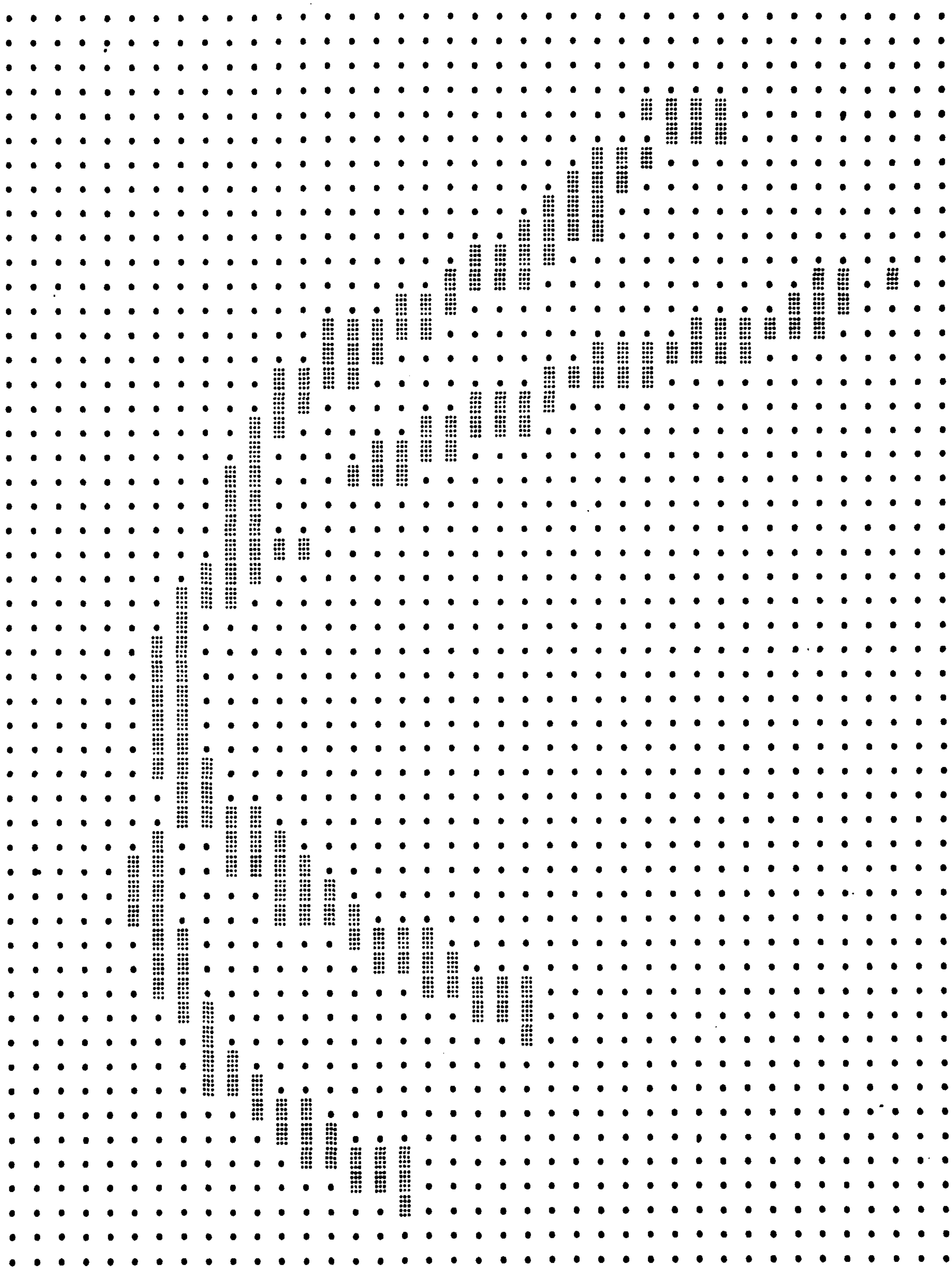


Figure 4.8. Gray-Weighted Skeleton of Human Chromosome; n=2; K=0.80 (from [23].

gray weighted MAT's, but the alterations are obvious, since the only thing that is changed is to compute the distances in accordance with the metric defined in (11) instead of that defined in the explanation of (3a) and (3b).

One thing that needs to be made clear at this point is that the b_{ij} are taken from the previously computed image at each pass except the first, and that the a_{ij} are always taken from the original image. It is easy to see that if the a_{ij} and b_{ij} are always taken from the same image, then convergence, at least in the case of the step interpolating function (13), cannot occur in a finite number of steps (Zeno's paradox revisted!).

A Grayscale Thinning Algorithm

Dyer and Rosenfeld [36] have devised the following algorithm for thinning grayscale images. The algorithm is based upon a grayscale scheme in which 0 is the maximum intensity value, i.e., the larger the grayscale value, the darker the pixel where a white pixel is assigned value 0.

Using the pixel neighborhood defined above, Dyer and Rosenfeld define the range, R, in the neighborhood as

$$R = \max(B,D,E,F,H) - \min(B,D,E,F,H) + 1 \qquad (15)$$

and choose an appropriate fraction, f, $(0 \leq f \leq 1)$ such that $[fR] = r'$, $0 \leq R \leq R$. Then the conditions which must be met for replacement of the element under E are:

(1) at least two of B, D, F, H have values $\leq E - R'$;

(2) for each of the six pairs of B, D, F, H, let m be the pair's minimum; then either E   m - R' or there is a path on the perimeter of the mask, i.e., not involving E, such that for every point p on the path, $p \leq m - R'$.

Iff conditions (1) and (2) are met, then E is replaced by the minimum of itself and 2 members of B, D, F, H. The choice of the 2 members of B, D, F, H is what determines the relationships of the objects in the thinned image to the objects in the original image. Dyer and Rosenfeld leave this choice to the user. In the case at hand, it is desirable to skeletonize the objects in the image, so the choice is made as follows. Set a flag, then scan the image by sliding the mask over the image in normal raster scan beginning with element (2,2) under E and ending with element (m-1,n-1) under E. At each point perform tests (1) and (2). On the first pass through the image and on all succeeding odd passes, replace E with the minimum of B, D, E. On each even pass, replace E with the minimum of E, F, H. When E is replaced, the replacement is written into a copy image which is created at the beginning of each pass.

The tests (1) and (2) are made on the original image in the first pass, and on the result of the previous pass on all succeeding passes. When E is replaced, the flag is reset. The procedure stops when a pass has been made and no E is replaced, that is, the flag is found set at the end of a pass.

The performance of this algorithm is dependent upon the range of grayscale values in the image and upon the choice of threshold.

Computation of a Grayscale MAT by Thinning

To convert the grayscale thinning algorithm of Dyer and Rosenfeld to compute the MAT of a grayscale image, all that has to be done is to build a list as was done earlier in the case of the binary thinning algorithm. In this case, there are two ways of defining the skeleton points: in the first, an element is a member of the skeleton iff two of its neighbors B, D, F, H have values $\leq E - R'$; in the second, an element is a member of the skeleton iff two of its neighbors B, D, F, H are less than E. The pass number, chain code scheme, etc., are no longer important because grayscale images cannot be reconstructed from their MAT's.

The results of this algorithm are compared to the algorithm of Levi and Montanari in Figure 4.9.

Conclusions

The Medial Axis Transform (MAT) of discrete binary and grayscale images can be computed by two different techniques, i.e., thinning and distance measurement, both of which simulate the "prairie fire" concept. The results of the two techniques differ, and it appears that the distance measurement techniques requires fewer operations, that is, it converges in fewer iterations. Moreover, the skeletons from the distance measurement technique appear to be more complete.

The thinning technique corresponds to the $N = 0$ case for the distance measurement technique. Although it is possible to define more complex thinning algorithms, it does not appear to be a fruitful area. The capability of the distance measure-

0	0	0	0	0	0	0	0
0	1	1	1	1	0	0	0
0	1	2	3	4	0	0	0
0	1	2	4	5	5	0	0
0	1	1	0	4	7	7	0
0	0	0	0	5	6	0	0
0	0	0	0	6	6	0	0
0	0	0	0	0	6	0	0

A

0	0	0	0	0	0	0	0
0	.5	.5	.5	.5	0	0	0
0	.5	2	2.5	2	0	0	0
0	.5	2	2	5	2.5	0	0
0	.5	.5	0	2	7	3.5	0
0	0	0	0	2.5	3	0	0
0	0	0	0	3	3	0	0
0	0	0	0	0	0	0	0

B

0	0	0	0	0	0	0	0
0	X	0	0	X	0	0	0
0	0	X	X	X	0	0	0
0	0	X	X	X	X	0	0
0	X	0	0	X	X	X	0
0	0	0	0	X	X	0	0
0	0	0	0	X	X	0	0
0	0	0	0	0	0	0	0

C

0	0	0	0	0	0	0	0
0	X	0	0	X	0	0	0
0	0	X	X	X	0	0	0
0	0	X	0	0	X	0	0
0	X	X	0	X	X	X	0
0	0	0	0	X	X	0	0
0	0	0	0	0	X	0	0
0	0	0	0	0	0	0	0

D

0	0	0	0	0	0	0	0
0	X	0	0	X	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	X	X	0	0
0	X	X	0	0	X	X	0
0	0	0	0	0	X	0	0
0	0	0	0	X	X	0	0
0	0	0	0	0	0	0	0

E

Figure 4.9. Grayscale Skeletons; a) Original Image; b) Result of Levi and Montanari's Algorithm, Skeleton Points Underlined; c) the Skeleton from Levi and Montanari's Algorithm; d) Thinning Skeleton with F=.5,(2)< E Test; e) Thinning Algorithm 2 < E - R' Test.

ment technique to be extended to higher order is a definite advantage in cases where the fine structure of the skeleton becomes important. A possible example of this could be in identifying a certain chromosome which closely resembles other chromosomes, etc.

Based on these findings, it appears that the distance measurement technique is preferable to the thinning technique and that future work should concentrate on the application of the distance measuring technique to pattern classification.

CHAPTER V

LINE CODING TECHNIQUES AND APPLICATIONS TO AIRPORT PATTERN CLASSIFICATION

Another aspect of digital image processing that has received much attention in the recent literature is the subject of image coding. The underlying objective of image coding has been the representation of an image with as few binary digits (bits) as possible, under the constraint that the resulting image contains some minimum level of fidelity (also called minimizing distortion). This portion of the report surveys the subject of image coding in some detail, in addition to presenting several specific coding schemes and illustrating each with an example.

Before presenting the two categories (or classes) of image coding schemes that will be covered in this report, it is necessary to define the meaning of the word "image." Throughout this coding discussion, the word "image" is to be taken as being synonymous with the words "digitized (i.e., quantized) line drawing." A line drawing is in turn defined as a two-dimensional representation of a digitized two-dimensional image in which only the boundaries (between each region in the image that possesses a constant gray level) are drawn (or "traced"). In other words, a line drawing merely preserves the boundaries formed by the different gray levels that appear within a quantized image.

For the purpose of the subsequent coding discussion, then, an image is taken to mean a two-dimensional line drawing that has undergone a quantization process. The degree (or "fineness") of this process is an important parameter in any digital image processing system, as it influences both the computer storage and processing requirements of such a system. For if an image is quantized very finely, then a large amount of computer memory and processing will be required; whereas, if an image is quantized too coarsely, computer memory and processing will be reduced, at the expense of a loss in image detail. Thus, such quantization should be as coarse as possible, while preserving those features of the original image that have been deemed significant [44]. The interested reader is referred to references [40] and [44] for a more detailed discussion of the quantization of continuous line drawings.

This portion of the report presents a survey of several schemes for the encoding of quantized line drawings. While the impetus for this discussion of image encoding was at least partially explained in the opening paragraph of this section, the reasoning behind the subsequent definition of an image as a quantized line drawing may not as yet be apparent to the reader. First of all, given that any modern image processing system employs one or more digital computers, then some analog-to-digital (A/D) process is required in order to convert the original analog (i.e., continuous-tone) image into its equivalent digital (i.e., discrete) representation. This A/D process is referred to in the literature as quantization. Secondly, many image

processing systems seek to recognize and classify particular objects (whether natural or man-made) of interest to the user. This latter process is referred to as pattern recognition in the literature (e.g., references [11], [18], [31]. From the point of view of pattern recognition via computer, line drawings have represented an important category of image representation [44]. It is for these reasons that we focus on the encoding of quantized line drawings.

Coding Categories

The various image coding schemes that will be discussed in this report can be divided into two categories:

- Line-by-Line Encoding
- Two-Dimensional Encoding

The first category, that of line-by-line encoding, involves the encoding of a digitized image on a per-scan-line basis. In other words, any encoding scheme within this category operates on each scan line independently of all others in the image. The following line-by-line encoding schemes will be discussed subsequently:

- Run-Length Encoding
- Differential Encoding

The second category, that of two-dimensional encoding, contains less restrictive encoding schemes in which each scan line is not encoded independently, but rather as a function of one or more of its adjacent scan lines. Thus, this latter category, unlike the first, inherently assumes that a given digitized image cannot be considered to consist of (either horizontal or

vertical) scan lines that are independent of each other. The following two-dimensional encoding schemes will be detailed later in this report:

- Chain Encoding
- Chain-Difference Encoding
- Line Adjacency Graph Encoding
- Region Adjacency Graph Encoding

A detailed discussion of and an illustrative example for each of the aforementioned encoding schemes are presented in the following paragraphs.

One-Dimensional Methods

Run-length Coding

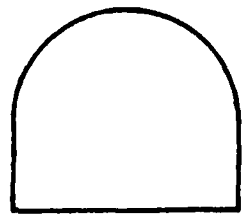
As mentioned previously, run-length encoding falls under the category of line-by-line encoding. Run length encoding is a relatively simple scheme in which the amplitudes of adjacent image points along a scan line are compared [30]. The run length is the number of juxtaposed image elements along a scan line having the same amplitude (i.e., gray level). Thus, for a given scan line whose image points are denoted X_1, X_2, $X_3, \ldots, X_N$, this scheme maps this sequence into a sequence of integer pairs (g_k, ℓ_k), where ℓ_k equals the number of consecutive image points of amplitude g_k.

The details regarding run length encoding are described in [47] and summarized as follows. For the first scan line of an image, the amplitude of its leftmost pixel is set equal to g_1 and ℓ_1 is set equal to the length of the run of points with this same amplitude g_1. Then at the first amplitude transition,

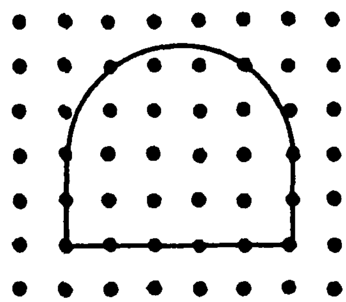
g_2 is set equal to the amplitude of the second run, with ℓ_2 equal to the length of this second run. This procedure is repeated until the end of the first scan line, after which succeeding scan lines are similarly encoded. It is evident then that, for any given scan line, the number of runs can vary from one (in which case every pixel has the same amplitude) to N (when no two adjacent points have the same amplitude). Correspondingly, the run lengths ℓ_k can vary from N to one. An illustrative example of run length encoding is presented in Figure 5.1.

Differential Coding

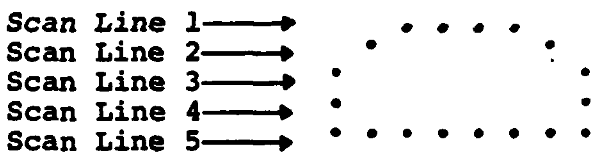
Another line-by-line encoding scheme mentioned earlier was that of differential encoding. The motivation for this scheme is the fact that since adjacent image points along a scan line are highly correlated in most pictures, there exists a high degree of redundancy between such pairs of points [30]. In differential encoding, only the differences in amplitudes of adjacent image points are encoded along a scan line, after having encoded the actual amplitude of the first point in that line. Thus, only the relative amplitude of each successive point after the first point in each scan is encoded in this scheme. However, a little thought would reveal that there is no real difference between run length encoding and differential encoding in terms of the encoding of quantized line drawings, as their points can be one of only two values: black or white. Thus, differential encoding will not be discussed further in this report.



a) Continuous Line Drawing



b) Method of quantizing line drawing shown in a)



c) Quantized line drawing corresponds to a)

For Scan Line 1: (0,2), (1,4), (0,2)
For Scan Line 2: (0,1), (1,1),)0,4), (1,1), (0,1)
For Scan Line 3: (1,1), (0,6), (1,1)
For Scan Line 4: (1,1), (0,6), (1,1)
For Scan LIne 5: (1,8)

d) Code resulting from the Run Length Encoding of c)

Figure 5.1. Continuous Line Drawing and its Quantized Regenerations

Two-Dimensional Methods

Just as it was true for the line-by-line encoding schemes discussed above, there are similarities between the four two-dimensional encoding schemes that will be covered in this report. Associated with two of these schemes (namely chain encoding and chain-difference encoding) is a particular quantization scheme, called grid-intersect quantization [45], that will first be described in the following paragraph.

Imagine that a uniform, rectangular grid is first superimposed on the continuous line drawing that is to be encoded. A quantized version of this line drawing can then be derived via a list of the coordinates of those grid intersections which come closest to lying on the continuous line drawing [50]. Thus, one representation of this line drawing is simply the list of these "line drawing points." However, it is not necessary to store this entire list of X-Y coordinates. Instead, each successive point in this list can be encoded relative to its previous point. The two aforementioned two-dimensional encoding schemes are two specific methods of performing this encoding, and each scheme is described below.

Chain Coding

Assuming that a given continuous line drawing has been quantized via the grid-intersect quantization method, then the chain encoding of the resulting quantized line drawing proceeds as follows. First, a point is chosen as the "starting point." The line segment from this starting point to the next point in the quantized line drawing can then be uniquely described (due

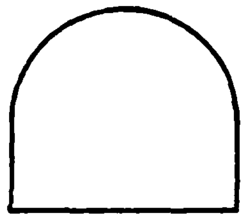
to the grid-intersect quantization) by the angle (measured clockwise) that this line segment forms with the positive x-axis. Thus, the chain encoding scheme represents this segment (and every successive line segment that makes up the quantized line drawing) by one of the following eight single-digit numbers:

Chain-Encoded Representation	Angle Between Segment and x-axis
0	0°
1	45°
2	90°
3	135°
4	180°
5	225°
6	270°
7	315°

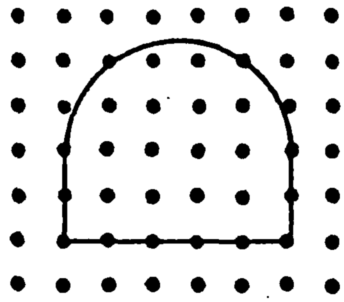
An example that illustrates both the grid-intersect quantization and this chain encoding scheme is shown in Figure 5.2, and further discussion of this scheme can be found in [43], [45], and [50].

Chain-Difference Coding

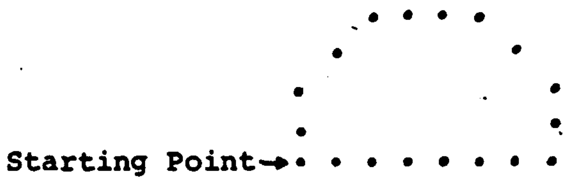
Chain-difference encoding is a modified version of chain encoding in which the difference in the angle between successive line segments is encoded (i.e., the relative angle between any two adjoining segments), rather than its actual value. Chain-difference encoding makes sense because, assuming that the quantization on the continuous line drawing was sufficiently fine so as to preserve all the detail of interest, then it follows that the (relative) difference in angle between adjoining line segments will ordinarily not exceed $\pm 45°$, whereas turns of $\pm 90°$ will be infrequent, and those exceeding 90° will



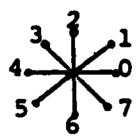
a) Continuous Line Drawing



b) Quantizing a) via Grid-Intersect Quantization



c) Quantized line Drawing corresponding to a)



d) Chain Encoding Scheme to be utilized on c)

2,2,1,1,0,0,0,7,7,6,6,4,4,4,4,4,4,4

e) Code resulting from chain encoding c) via d)

Figure 5.2. Grid-Intersect Quantization and Chain Encoding

be fairly rare [45]. Chain-difference encoding takes advantage of this information by representing the small angle differences by a short code, while using successively longer codes to represent the larger possible angle differences (analogous to Huffman Codes, which are detailed in [46]. Thus, it can be shown that, in general, chain-difference encoding requires only slightly more than two-thirds of the storage required by chain encoding [45].

Both chain encoding and chain difference encoding are two of the two-dimensional encoding schemes in which the input image is basically considered to be a linear and ordered list of elements [50]. But another methodology for encoding continous line drawings is to consider them as a set of interrelated objects or regions. The remaining two encoding schemes, that of line adjacency graph encoding and region adjacency graph encoding, implement this methodology. Both of these schemes are detailed in [50] from which the following discussion is taken.

Line Adjacency Graph

A line drawing consists of black lines on a white background (or vice-versa). The line adjacency graph encoding scheme is similar to the run-length encoding scheme described previously in that both schemes successively scan each line that constitutes the quantized line drawing in order to record where within each scan line there is a change (called a break point) from black to white or white to black. However, the line adjacency graph encoding scheme partitions each scan line into segments at these break points. As the line drawing is

scanned, a graph structure called a line adjacency graph is constructed as follows. The "nodes" of this graph consist of those segments (or individual points) that constitute a part of the line drawing. Any two nodes of this graph are then "connected" if the following conditions are satisfied:

(1) If the segments (or points) represented by the two nodes lie on adjacent scan lines,

(2) If the segments (or points) also overlap when projected onto a single line.

An illustrative example of this scheme is shown in Figure 5.3

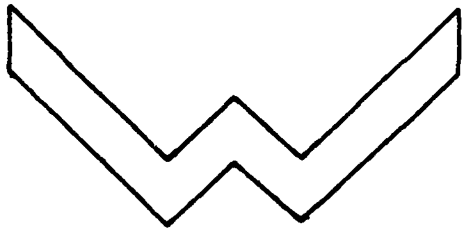
It is evident from the preceding discussion that the usefulness of this line adjacency encoding scheme lies in the fact that it retains the two-dimensional structure of the line drawing. Thus, this scheme is more powerful than any line-by-line encoding scheme, which preserves left-to-right relations but not up-and-down relations.

Region Adjacency Graph

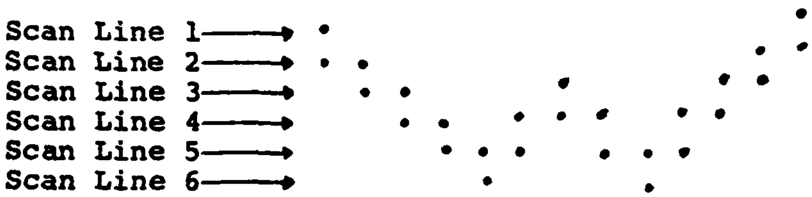
A related structure that is also useful in segmentation of a line drawing is called the region adjacency graph [50]. In this type of graph, the nodes represent regions shown in the line drawing, and two nodes are connected if the regions represented by these nodes are adjacent. An illustrative example of this region adjacency graph encoding scheme is presented in Figure 5.4.

Comparison of Types

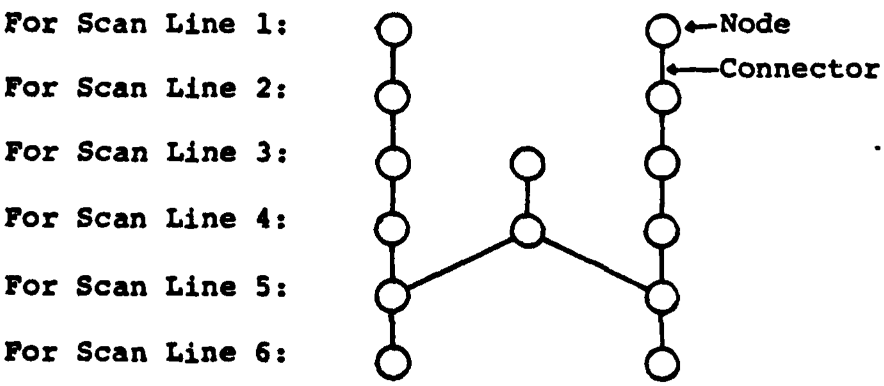
At this point the reader should not only be familiar with several specific image encoding schemes, but should also have



a) Continuous line drawing of the letter W.

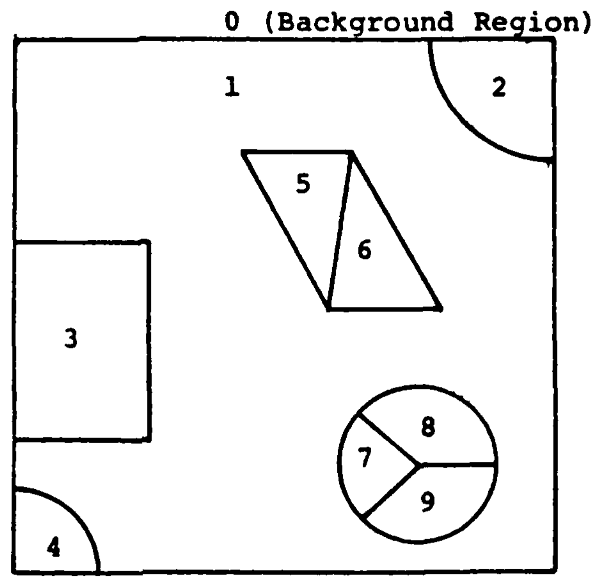


b) Quantized representation of a)

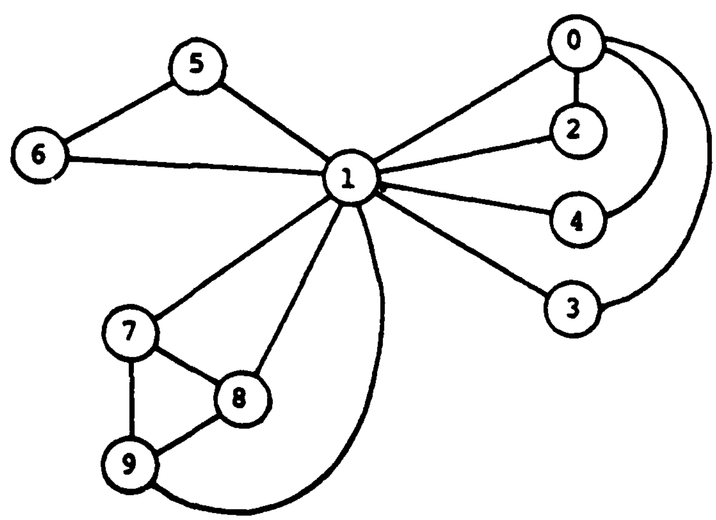


c) Line Adajacency Graph Encoding of b)

Figure 5.3. Line-Adjacency Encoding



a) Line drawing containing ten regions



b) Region adjacency graph encoding corresponding to a)

Figure 5.4. Region-Adjacency Encoding

AD-A091 600 GEORGE WASHINGTON UNIV WASHINGTON DC DEPT OF ELECTRI--ETC F/G 9/2
ANALYSIS AND DEVELOPMENT OF IMAGE STATISTICS AND REDUNDANCY REM--ETC(U)
SEP 80 M H LOEW, R L PICKHOLTZ, L GOLDMAN DAAK70-79-C-0147
UNCLASSIFIED GWU-EE-CS-80-09 ETL-0239 NL

2 - 2
AD
A091 600

END
DATE
FILMED
12 80
DTIC

a better insight into why these schemes were divided into two fairly distinct categories: line-by-line encoding vs two-dimensional encoding. So now an interesting question arises: Does either of these categories have any inherent advantages over the other? This question is addresssed both from a practical and a more theoretical aspect in the following paragraphs.

From a practical standpoint, those encoding schemes that fall under the category of line-by-line encoding are superior to those that were categorized under two-dimensional encoding. The reason for this lies in the fact that, almost invariably, one of the first steps that any digital image processing system implements is to convert an input two-dimensional image into a series of independent scan lines (either horizontal or vertical). Subsequent encoding of each of these independent scan lines via a particular line-by-line encoding scheme yields a considerable reduction in computer processing complexity and storage requirements over those of two-dimensional encoding [49].

Before comparing these two categories of image encoding schemes from a more theoretical standpoint, a brief discussion of rate distortion theory is necessary. This theory centers around a concept called the rate distortion function, which will be introduced via the simplified block diagram of a communications system shown in Figure 5.5. The following development is taken from [41].

Rate Distortion Theory

Given the source (e.g., an image) shown in Figure 5.5, the communications systems engineer is confronted with the

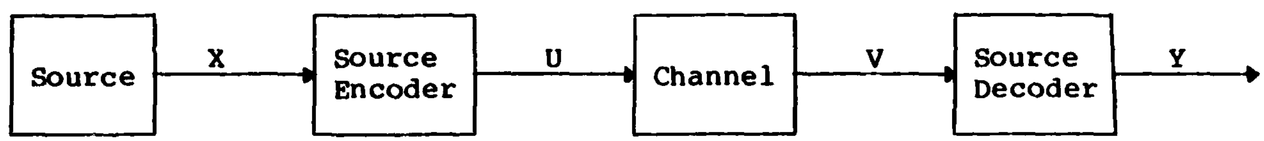


Figure 5.5. Communications System Block Diagram

problem of encoding this source in such a way that the channel capacity requirement for transmission is minimized. (The "capacity" of the channel of Figure 5.5 will be defined shortly). To achieve this, he is willing to tolerate some average distortion between the source output and the decoder output. The problem addressed by the subject of rate distortion theory is the minimization of this channel capacity requirement while maintaining this average distortion at or below an acceptable level (as defined by the user).

To be more specific, let the average information transmitted from the source output (labeled X in Figure 5.5) to the decoder's output (labeled by Y in this figure) be denoted by a function I(X,Y). Further, assume that more information corresponds to larger values of this function I(X,Y). Now let the analogous average information transmitted from the source encoder output (labeled U) to the channel output (labeled V) be denoted I(U,V). The "capacity" herein denoted by C of the channel shown in Figure 5.5 can now be defined as the maximum of I(U,V) over all possible input devices. It is then evident via Figure 5.5 that the intervening nature of U and V implies that:

$$I(X,Y) \leq I(U,V) \leq C$$

Establishing that it is possible for I(X,Y) to be arbitrarily close to C for an arbitrary source and channel is not so easily performed, and the interested reader is referred to references [39], [41], [46], and [49] for further details. Suffice it to

say that the rate distortion function, usually denoted R(D), is the minimum value of I(X,Y) for a given distortion level D, and only for values of R(D) less than C can the communications engineer be insured of obtaining a distortion level D with the system of Figure 5.5. In other words, this rate distortion function R(D) is the minimum source encoder output rate, and hence minimum channel capacity, required for a given average distortion level D.

The impetus for this introduction of the rate distortion function will now become apparent. It should be evident from the above discussion that a rate distortion function could be derived for a system of Figure 5.5 employing a line-by-line source encoding scheme. Similarly, another rate distortion function could be derived for an identical system employing a two-dimensional source encoding scheme. A comparison of the resulting rate distortion functions could then be utilized to determine which of these two schemes has a theoretical advantage over the other.

Generalized rate distortion functions for both the line-by-line encoding and two-dimensional encoding of a particular two-dimensional image (that is homogeneous and isotropic) are each derived in [49]. However, the resulting pair of rate distortion functions is found to be too complex to permit a direct comparison between them. Thus, [49] proceeds to numerically evaluate these two expressions for a particular example. The results of this evaluation can be summarized [49] as follows:

(1) Two-dimensional encoding permits a decrease in coding rate by a factor of 2 to 3 over that of line-by-line encoding, and

(2) In line-by-line encoding, the selection of the line-width between consecutive scan lines is crucial in achieving an efficient coding rate.

While the authors of reference [49] achieved these results via a number of simplifying assumptions, they feel that their conclusions (stated above) are indicative of the differences for these two categories of encoding complexity. Furthermore, a later and independent evaluation by another author was in approximate agreement with their first conclusion above (see [41]). Thus, from a theoretical standpoint, it appears that any two-dimensional encoding scheme is superior to any line-by--line encoding scheme.

Line Drawing Encoding - An Airport Example

Introduction

The notion that precipitated the interest in image coding was the proposition that, given a complex image, there exists some series of preprocessing followed by suitable encoding that may result in a symbol stream that exhibits a regonizable pattern characteristic of an identifiable object. The system proposed in the notion is shown in Figure 5.6. A variety of image encoding schemes has been proposed in the literature. Some of these are listed in Table 5.1. Other techniques that are often used in the image processing context, some or which may be considered an encoding techniques, are shown in Table 5.2.

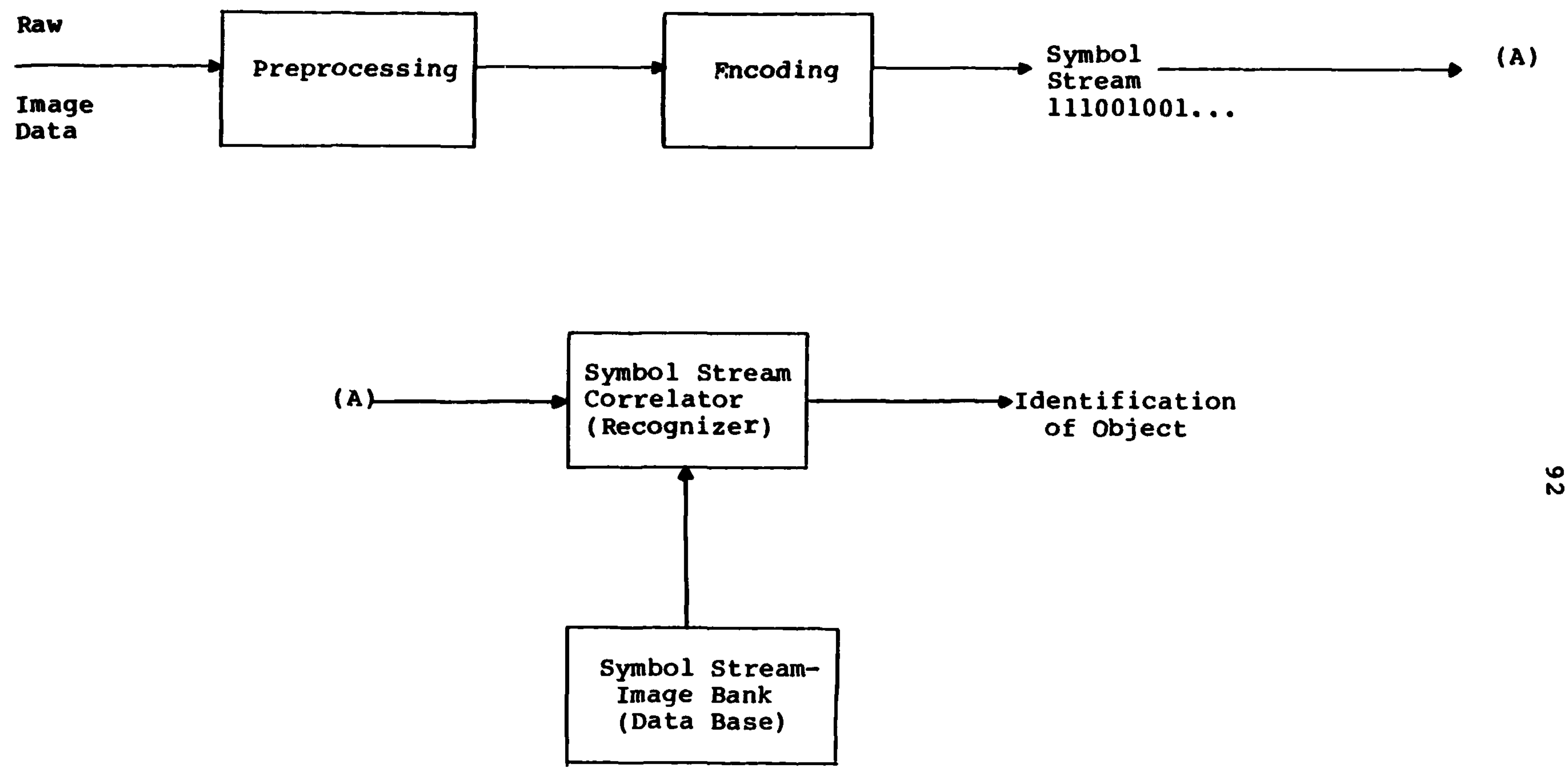


Figure 5.6. Image Classification Scheme

Table 5.1

A Sampling of Available Image Encoding Techniques[1]

Error-Free Encoding Schemes

- Difference Encoding with Shift Coding
- Contour Encoding
- Gray Code
- Huffman Code
- Shift Code
- Run-Length Encoding
  - One-Dimensional
  - Two-Dimensional

Error-Friendly Encoding

- Differential Pulse Code Modulation (DPCM)
- Transform Encoding
- Hybrid Encoding

- Hotelling Transformation

[1](From [47]).

Table 5.2

Symbolic Image Description Techniques

- Connectivity (Pixel Connectivity)
- Line Description
  - Curve Fitting
  - Line-to-Point Transformation
- Shape Description

- Metric Attributes	Area Length-to-Width Ratio Perimeter Perimeter to Area Ratio
- Topological Attributes	Euler Number Connectivity
- Analytical Attributes	Polar/Fourier Curvature Function Moment Approximation

Associated Tools and Techniques:

- Shrinking, Thinning, and Skeletonizing
- Amplitude Segmentation
  - Luminance Thresholding
  - Multidimensional Thresholding (Color)
  - Regional Growing
- Edge Segmentation
  - Curve Fitting
  - Contour Following
  - Edge Point Linking
- Texture Segmentation
- Shape Segmentation

Encoding as an Identification Tool

Used in its most familiar context, encoding is a process whereby information in one format is converted into another format as a necessary step in some transmission or storage process. In this context the objective of the encoding scheme, as noted earlier, is to convey the original information content with minimum loss of information and added extraneous information, while using the minimum number of encoding symbols. In the image identification problem the objective is quite different. Here the goal is to choose an encoding scheme that preserves only enough information to identify those key attributes that uniquely identify the target or interest. "How do the two situations differ?", one may ask. In the one case (the transmission or data storage example) the observer may be required to distinguish details or attributes that cannot be established a priori. Such an example may be a <u>specific</u> human face as opposed to a general decision that a human face is indeed being displayed. In the latter case, as in this study, the objective is to identify an object that belongs to a specific <u>class</u> of objects. While the individual members of the class may exhibit characteristics that set themselves far apart from other members of the class, they all share one or more common characteristics that we use to combine them in description as a common entity. <u>These</u> characteristics are the <u>only</u> characteristics necessary to encode if an encoding scheme is indeed to be useful as a tool in the image identification process. In addition, target information that merely details the class member or simply makes

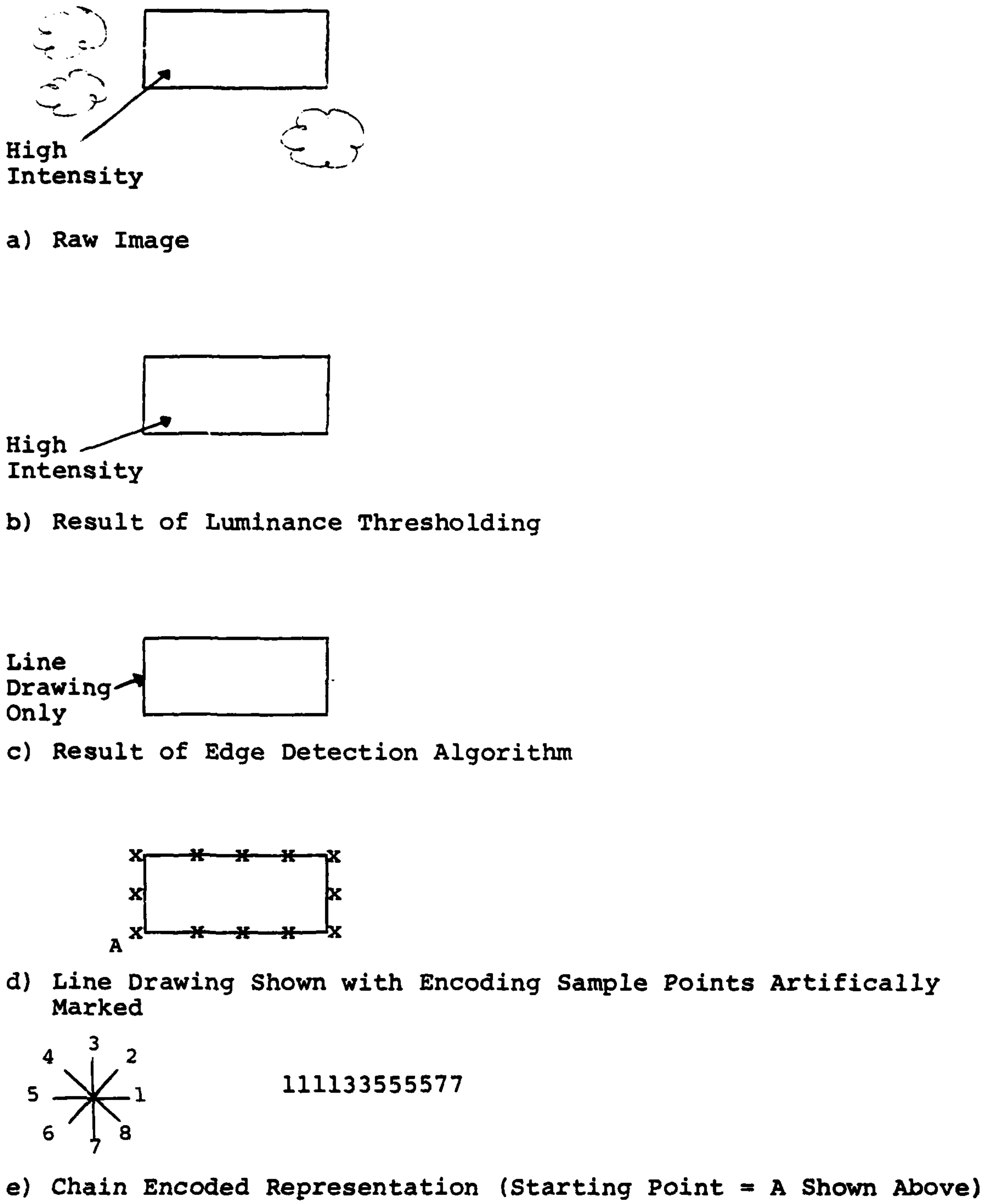
it somewhat different from another member of the class is extraneous to the target identification process. In the encoded form the resultant encoded version of the image may therefore contain information extraneous to the target identification process. This information will serve only to increase the processing required to complete the pattern recognition process. The objective, of course, is to choose a preprocessing scheme and particularly an encoding algorithm that suppresses the noise (image detail) and retains or emphasizes only that information relative to the attribute or attributes that uniquely identify the target as a member of the target class of interest.

Of course some encoding schemes may be appropriate to the situation described above. It is proposed here that the validity of the proposition is a function of the nature of the class and the level of specificity of the definition of the class. Two specific example targets will be proposed in the following discussion. In both cases Freeman chain encoding will be proposed as an encoding scheme. In the first case, the scheme appears to be an appropriate and quite valuable technique to aid in a straightforward implementation of the identification process of Figure 5.6. In the second case, quite the opposite is true due to the level of variation of the members of the class.

In one proposed scheme for target identification perhaps one or more of the tools and techniques like those identified in Table 5.2 would be required to preprocess an image to develop a candidate subimage. The candidate subimage would perhaps be further preprocessed by applying an edge-detection algorithm.

All of these preprocessing steps would be carried out sequentially in the first process shown in Figure 5.6. In the subsequent encoding process shown in that Figure a suitable encoding technique would be applied to the preprocessed image. In the first example, suppose that the raw image plane contains the top view of a small isolated building as a high illuminance rectangle of relatively small size contrasted against a dark background representing a highly vegetated region. Preprocessing by luminance thresholding followed by edge detection would reveal an image composed of a rectangle line drawing. Encoding this line drawing by either a raster scan or Freeman encoding technique would result in a relatively simple symbolic description of the rectangular shape. Figure 5.7 illustrates the Freeman chain-encoded symbol stream. Based directly on the symbol stream both the shape (rectangular) and area can be determined directly, as shown in the final process of Figure 5.6. If these attributes are sufficient to imply an isolated building structure (based perhaps on a priori knowledge of the types of targets expected in the image) then the small rectangular area can be positively identified immediately as an isolated building structure.

Figure 5.8 illustrates a subimage obtained from one of the aerial photographs made available under the research contract. Superimposed over the photograph is a clear rectangular grid used in manually performed encoding operations on the image. Figure 5.9 is an expanded view of the same scene, but it represents the result of preprocessing steps that isolate the large



a) Raw Image

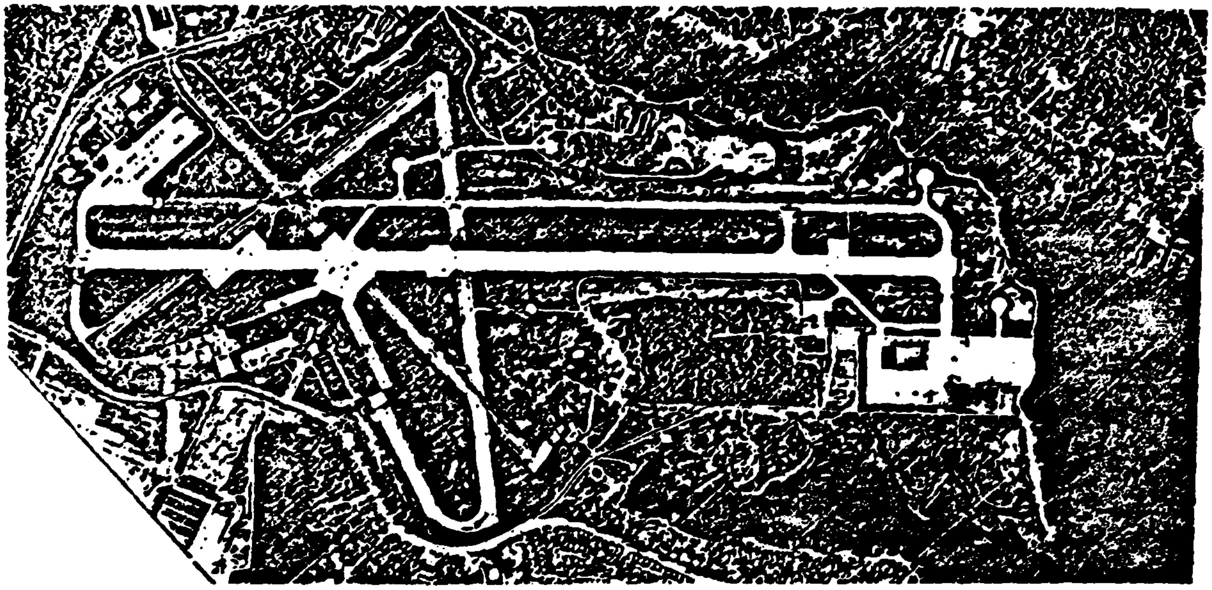
b) Result of Luminance Thresholding

c) Result of Edge Detection Algorithm

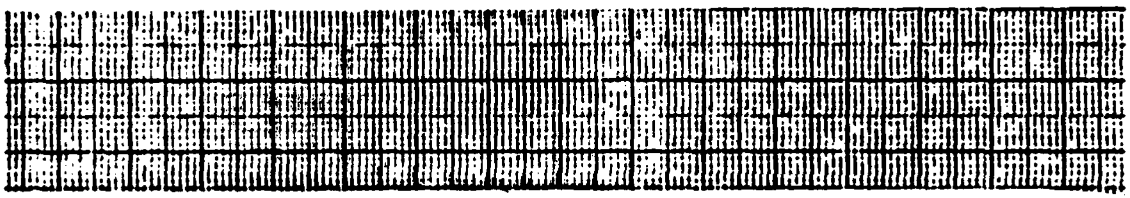
d) Line Drawing Shown with Encoding Sample Points Artifically Marked

e) Chain Encoded Representation (Starting Point = A Shown Above)

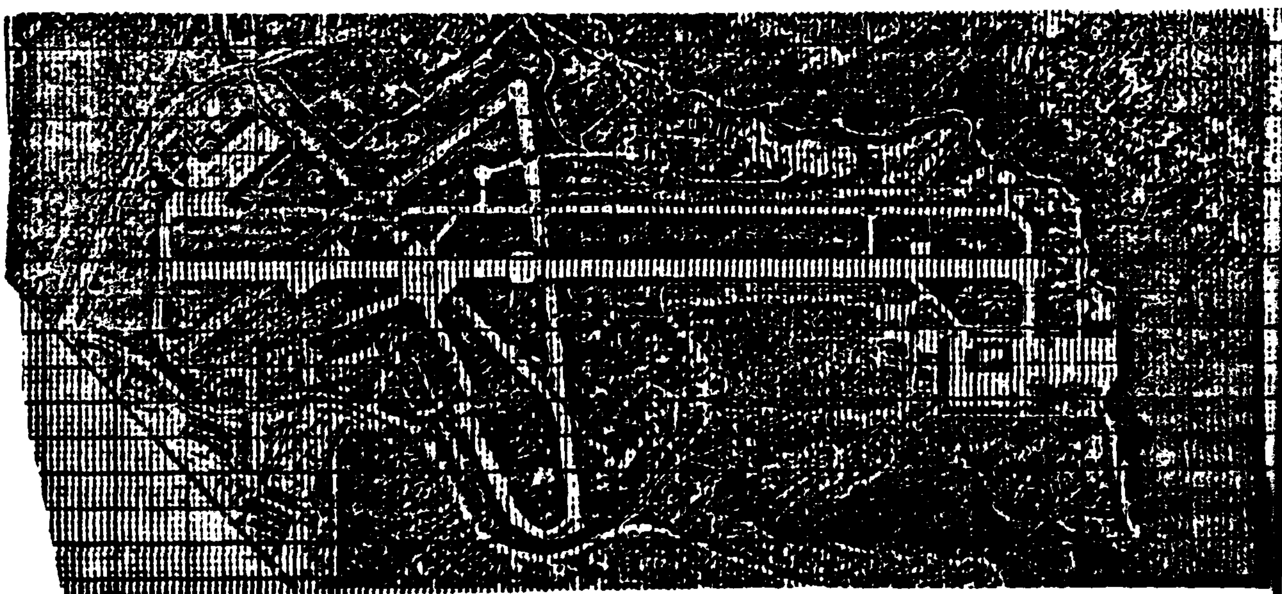
Figure 5.7. Chain Encoded Image Stream



a) Aerial Photograph Containing Airport Sample Image



b) Rectangular Overlay Representing Hypothetical Pixel Array



c) Aerial Photograph With Hypothetical Pixel Array

Figure 5.8. A Subimage Ready for Encoding (Airport)

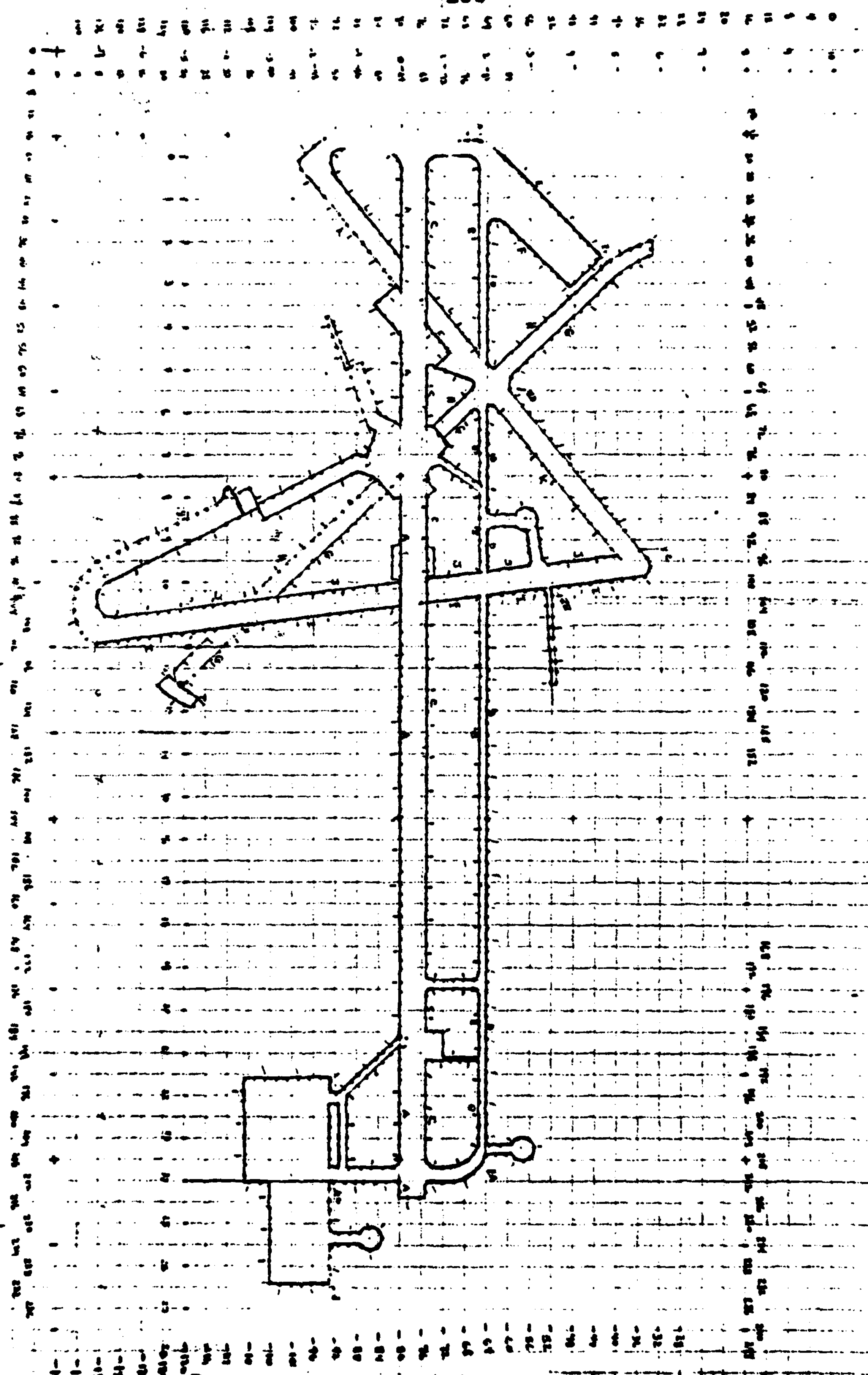


FIGURE 5.9. EXPANDED VIEW OF THE SUBIMAGE

complex high intensity structure represented by the runway, service road, ramp pattern followed by an edge detection algorithm that limits the product image to a line-drawing output. Here the high intensity pixels representing the outline of the airport geometry would be displayed on a dark background. Shown also in the Figure are "tick-marks" along the line drawing that represent sampling points where encoding will be performed. These "tick-marks" were placed entirely randomly here, but in a realistic situation the sample points would be adjacent pixels in a rectangular array. As a result, there would immediately be introduced a measure of irregularity in an otherwise straight line due to the finite pixel size in the image. Therefore, the sample points shown are only for demonstration purposes.

The first encoding scheme applied to the line drawing sample points is a raster scan code. In this encoding process the X and Y coordinates of high intensity pixels are merely read sequentially into a vector. The vector may be either used to reconstruct the line drawing at a receiving site, or in this case it may be used in a pattern-analysis to determine its shape or some other key attribute. Table 5.3 illustrates the raster scan result. Here only every fourth point was encoded.

In a second effort the same sample airport line drawing geometry was chain encoded using the Freeman encoding scheme with eight levels of angle encoding. Shown in Table 5.4 is a partial symbol output vector. The vector may be read by simply linking subsequent columns in the table. Several symbols appearing in the table should be defined for the reader. First, the

Table 5.3

Raster Scan Encoding of Sample Airport Geometry

	X	Y_1	Y_2	Y_3	Y_4	Y_5			
1	33	36	38						
2	34	36	38	96	97				
3	35	36	39	95	98				
4	36	37	40	93	99				
5	37	37	40						
6	38								
7	39								
8	40	38	42	87	93	95	99		
9	41								
10	42								
11	43								
12	44	36	40	43	47	82	88	95	100
13	45								
14	46								
15	47								
16	48	33	44	46	50	77	84	95	100
17	49								
18	50								
19	51								
20	52	29	41	50	54	72	79	96	104
21	53								
22	54								
23	55								
24	56	25	36	54	58	68	74	86	90
25	57								
26	58								
27	59								
28	60	20	32	58	69	87	89	97	100
29	61								
30	62								
31	63								

Table 5.4

Chain Encoding of Periphery of Airport Line Drawing

	1	2	3	4	5	6	7
1	A	5	7	6	5	4 ZA	6
2	7	6	8	4	M	3	M
3	M	5	7	M	2	3	19 ZD
4	13	6	8	7 U	3	2	5
5	8	1	7	3	M	3	4
6	M	2	8	M	2	4	M
7	2	1	7	14	2	5	16
8	1	2	8 S	1	4	6	5
9	2	1	8	M	S	7	5
10	M	2	8	38	S	8	7
11	12	1	1	8	7	7	8
12	8	2	M	M	8	7	M
13	M	1	3	7	M	5	16
14	2	2	2	6	2	M	6
15	2	1	M	5	5	50	5
16	M	1	2	7	M	3	M
17	2	1 q	3	M	3	M	12
18	1	8	M	7	4 ZZ	4	4
19	M	7	10	1	3	1	3
20	7	8	8	M	M	M	2
21	8	M	M	8	5	7	M
22	6	8	4	2	1	3	7
23	6	6	1	3	2	5	3
24	5	5	6	3	3	M	4
25	6	8	8	1	4	7	M
26	5	7 R	2	M	4	4 ZB	2
27	6	8	M	3	3	3	5 zE
28	5	7	2	3	3	M	6
29	7	8	3	M	3	7	M
30	5	7	4 T	4 Z	4	4	13
31	6	8	6	4	4	5 Ze	AE

implementation of this algorithm assumes that a contour following algorithm is employed. Code symbols may therefore include stop and start points. "A" is a starting point and "AE" represents an end point. In addition, a run symbol "M" was included to reduce the length of the vector when multiples of a symbol are deduced by the encoder. The small letters occurring in the table refer to benchmark points that are shown in the original line drawing. These are included only for reference purposes. It is interesting to note that the chain-encoded portions of the vector representing the long straight sides of runway surfaces are interspersed in other segments that represent significant detail. If this attribute--"long ribbon, etc." is a key attribute, then a significant amount of processing of the encoded symbol string will likely be necessary to extract the desired attribute. Although this encoding scheme does not preclude further use in the context of identification, the structure is not such that the encoding emphasizes the key attribute identified above. If, on the other hand, the key attribute of interest is total high intensity (paved) area, then the chain encoding scheme may be considered appropriate, since area measure is very straightforward in the the chain encoded format.

Table 5.5 illustrates an output that would be obtained if Hough transformation were applied to the line drawing, followed by a threshold on the counts obtained in the quantization bins. Assumed here are fairly small levels of quantization. Shown in the Table are three numerical entries in each column. The column is headed by a letter that identifies the straight line in the original figure that resulted in the Table entry. The

Table 5.5

Output of Thresholded Hough Transform

A	B	C	D	E	F	G	H	I	J	K	L	M
81	80	80	79	15	10	42	46	49	44	38	39	14
$\angle$ +	$\angle$ +	$\angle$ +	$\angle$ +	$\angle$ +	$\angle$ +	$\angle$ +	$\angle$ +	$\angle$ +	$\angle$ +	$\angle$ +	$\angle$ +	$\angle$ +
0° 80	0° 65	0° 75	0° 64	44° 84	44° 98	310° -8	310° -4	277° -156	277° -152	34° 120	34° 115	87° +250

upper figure is the bin count, the number under the symbol is the angular bin number (angle of line) and the number under the symbol is the value of rho in the BIN (perpendicular distance from origin). The Table not only indicates the relatively long lines (high bin count), but it also indicates those of relatively equal length (possibly representing long ribbonlike structures--roads or runways). It also offers information, as noted in Chapter II, that will allow a computation of distance separating the lines (road width or runway width). For example, by merely consulting Table 5.5 the reader can immediately determine that lines A, B, C, and D are long and parallel. Furthermore, by observing the value of rho he can observe that lines A and C are separated widely (indicating a wide expanse--perhaps a runway) and that lines B and D are narrowly separated (indicating perhaps a narrow road or taxiway).

In the latter case the Hough routine is immediately recoggnized as a valuable encoding tool because it possesses the characteristic that it serves to emphasize a key property that is of importance in identifying the target of a particular class. Chain encoding, while of some value, would require additional processing subsequent to the encoding to extract the key attribute, and raster scan encoding would likely be of even lesser value.

The Airport Recognition Problem - Cues and Attributes

Introduction

An airport is a striking target, easily identified by the human observer in the type of aerial photograph available to this study. Because of this observation, it seemed that this

type of target structure apparently offered a rich set of cues or attributes with which to identify the nature of the target. The objective of this effort was to identify and document those attributes and to formulate a methodology, algorithms, and software to perform the identification process.

Before further researching the problem, several notions regarding the target geometry were intially considered as possible tools for use in the identification process. In some respects these tools or attributes appear to be valid across the board as several diverse airport structures are considered. Given the limited data base available, a more complete set of aerial data was obtained and studied to identify more accurate target attributes.

Much of the thrust of preceding efforts has used geometry as a primary means of sorting or identification. This thrust has influenced the current efforts in that geometry was considered again as a prime candidate in investigating the means necessary to perform the airport identification process. It appears, however, that other cues may also become extremely valuable in the identification process. A decision tree application to the airport decision problem will be identified and described in a subsequent discussion. One important attribute identified in the decision tree is the extremely large area, smooth, uniform intensity runway surface. Although not a positive means of identification, these attributes can increase the probability of a correct decision when coupled with other features.

Airport Characteristics - A Description

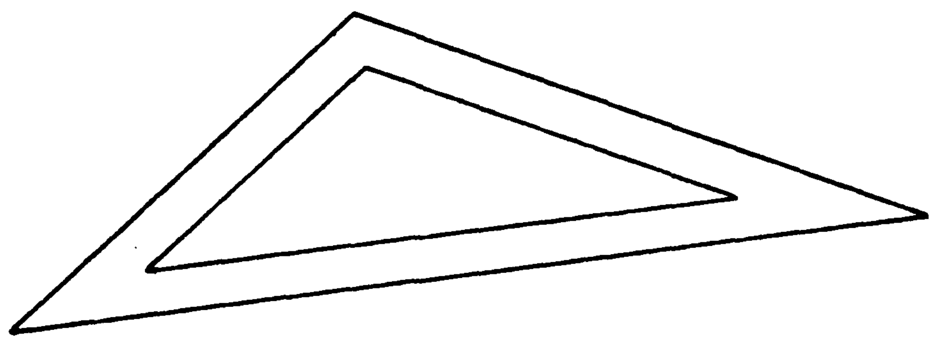
One method for identifying objects or "targets", as they will be called in this discussion, is to simply create a mask or template using a known geometry or some other known attribute. Such a technique is extremely applicable in the case of an optical character reader equipped to read characters from a fixed character font. In this case the target set is a fixed controlled set known a priori to the character reader. The primary problem for the character reader is then that it must distinguish <u>one character from another</u>. Generally, all of the targets confronting the character reader will come from the fixed character set, a well known, well characterized number of target objects.

In the current study the situation is somewhat different. Here the "reader" may confront any one or more of a large, almost unbounded, number of targets. Some of these targets may be from the list of "important" targets identified a priori to the reader, but other items may be unknown. Many target candidates may be large in scale, as in the case of the airport, and other candidate targets may be very small, as in the case of isolated building structures. Therefore, even the field of view and orientation with respect to the target in the airport identification will be unknown a priori.

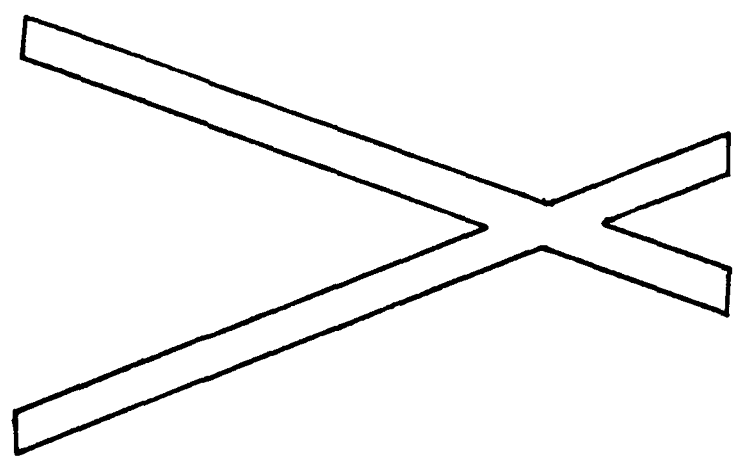
The airport identification problem--indeed, the problem of identifying most of the structures in the list of important targets--is further complicated in that the targets are generic or functional structures rather than specific descriptions that

possess specific characteristics. A simple target like a single isolated building, for example, may vary in both size and shape. It may even have a black roof or a highly reflective roof, yet it is said to belong to a single class called "detached buildings." The same is true of the airport structure. Some major airports will include several landing strips, parallel taxiways, and immense aprons, while others will be more compact. One proposition put forth in early discussions is that a basic structure is that depicted in Figure 5.10. One candidate in the Figure shows three large landing strips, roughly in a triangular pattern, crossing one another at random acute angles. The second shows a simple crossed-V shape. Such a geometry, if applicable, could be encoded and identified based on code symbol characteristics. The identification scheme, it was postulated, could be sufficiently general so as to accept variations in specific runway lengths and intersection angles.

Before continuing along these lines it was imperative that actual target characteristics be investigated so as to ascertain their specific nature. Such a procedure is necessary in the case of generic or functional targets because they take so many forms. The extraction of features (or attributes) thus becomes a necessity; template-matching is ruled out. Expressed in words, such a feature may be, for example, "long, wide, straight ribbons of concrete." This attribute must, of course, be capable of representation in machine form, and the words "long", and "wide" must be reduced to some numerical range that can be tested.



a) Candidate 1



b) Candidate 2

Figure 5.10. Proposed Model Airport Geometries

Before examining specific examples and identifying specific attributes, we should examine some ideas that were proposed in the context of the complete cartographic target set. This is important since a major objective of this program is to classify targets. In the discussion regarding the categorization of the targets into classes as a means of developing attributes, the airport structure was identified as a "complex" structure in that it incorporates road and highway-like structures and that it also incorporates buildings and parking facilities. Often major airports are located in relatively rural areas, although some examples can be cited where large cities include landing fields within their boundaries just adjacent to large urban concentrations. One method that could be used to identify an airport would be to identify its fundamental components (i.e., runways, taxiways, hangars, parked aircraft, parking lots, administrative and support buildings), then to make a decision based on whether or not the image contains a sufficient set of the above attributes as identified in a previous set of decision processes. Such a process is shown pictorially in Figure 5.11. It is fundamentally a process based on concepts of syntactic pattern recognition. For each of the fundamental structures runway, hangar, etc.) an identification procedure would be applied to the image. A syntax of geometrical constructs (angles, long straight sides, etc.) would be appropriate to identify some of the candidate components of the airport structure. Other structures would be identifiable based on intensity or texture measures. Then at the next higher level of decision-

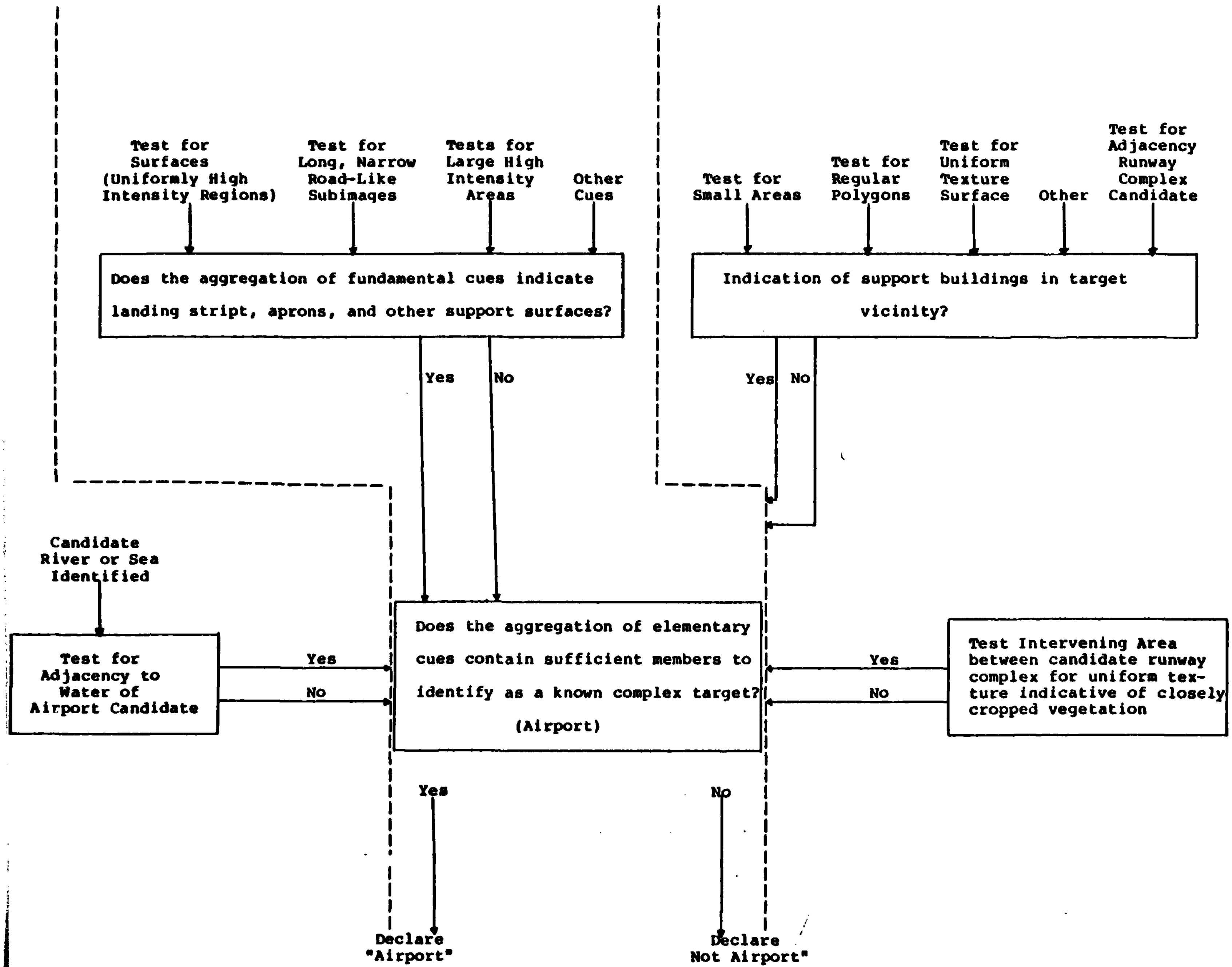


Figure 5.11. Decision Process for Airports

making, component descriptions would be used as inputs to the final decision regarding the complex structure. Such a bottom-up approach could also be used to identify urban areas, starting first by identifying component structures in an image, such as buildings and streets. This differs from the top-down approach where, for example, a texture measure may be used to determine if an image depicts an urban or a rural area simply by looking at the general textural measures of a large image and then directly concluding from it the nature of a complex target like an urban area.

The technique adopted as a result of this study may be regarded as a variant of the bottom-up approach in that several characteristic attributes of the airport structure have been identified, and these are used as intermediate decision points in the overall decision process. Other components that have not been currently considered in the logic tree structure for the airport problem have been what are considered secondary and tertiary cues like buildings and parking lots. Table 5.6 includes several cues that have been identified as characteristic of most airport structures. These have been developed based on observation of sample image characteristics and direct observation of airport geometries and characteristics from the air.

Figures 5.12 through 5.21 are line drawings of ten randomly selected airports from an official document published by the Air Traffic Service of the Federal Aviation Administration. The document provides major airport visual outlines and brief

Table 5.6

Characteristics of Airport Structures

- One or more (1-5) runways
- Runways are long but of limited (abrupt end) length
- Runways are extremely straight and uniform
- Most airports include secondary roadlike structures as aprons and taxiways
- Runway width often exceeds that of standard roads
- Airports are on relatively flat land
- Airports are accessible by roads
- Runways are concrete--white and highly reflective
- Most airports include buildings for airport administration and hangars
- Most major airports include large parking facilities
- Often the region enclosed by the runway and taxiway surfaces is of uniform texture, charactreristic of low cut vegetation to facilitate ground visbility

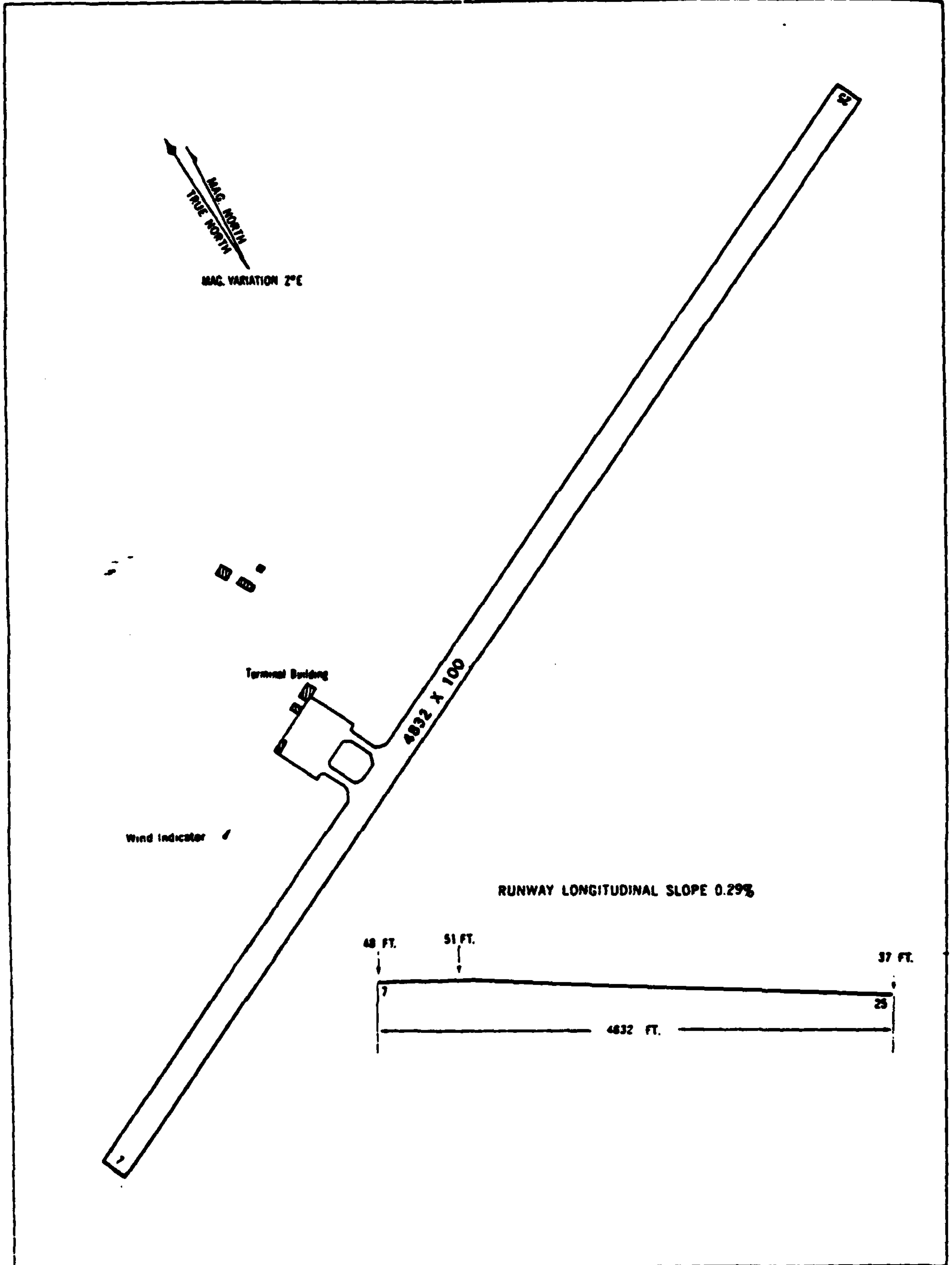


Figure 5.12. Yap

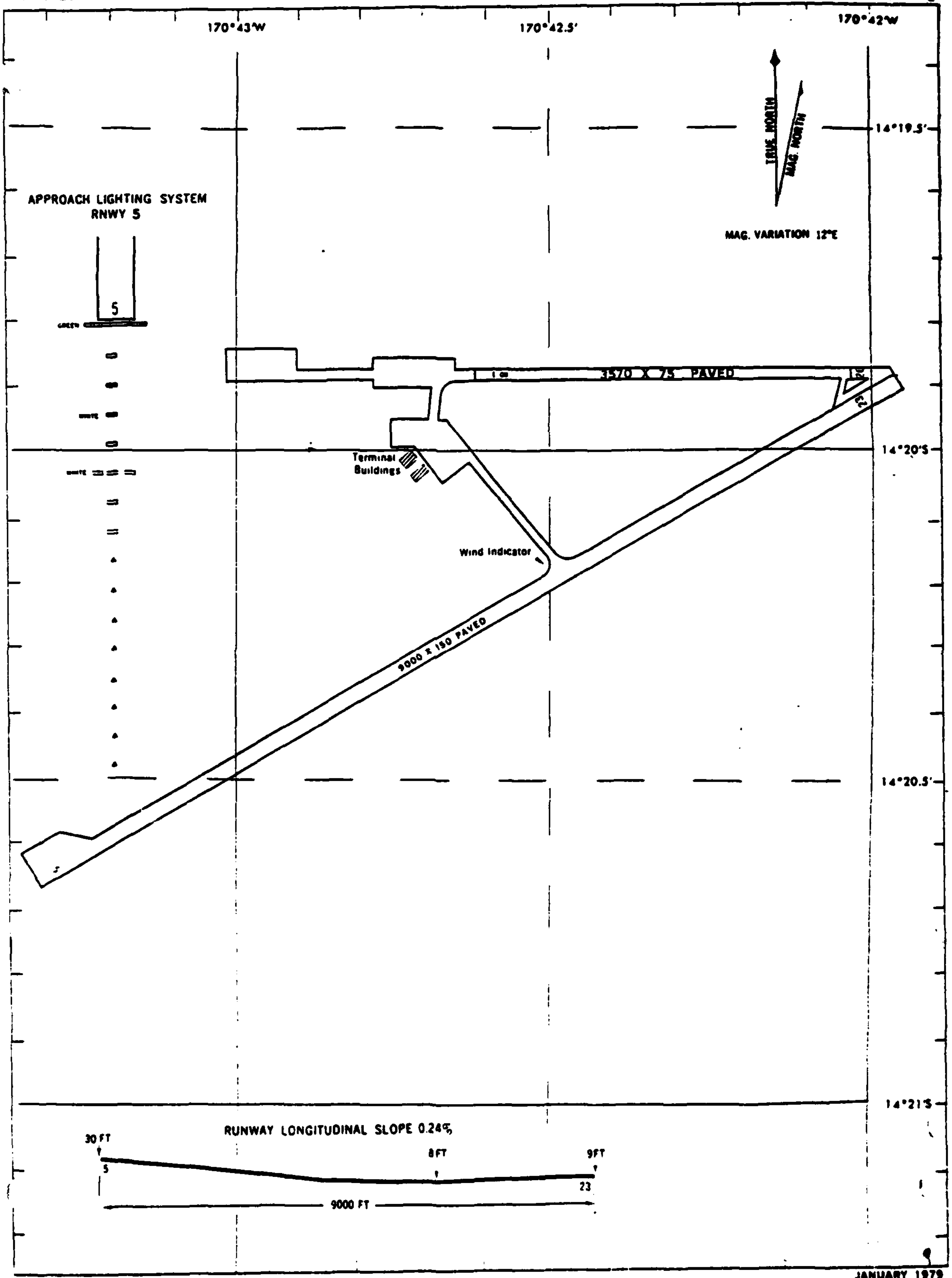


Figure 5.13. Pago Pago

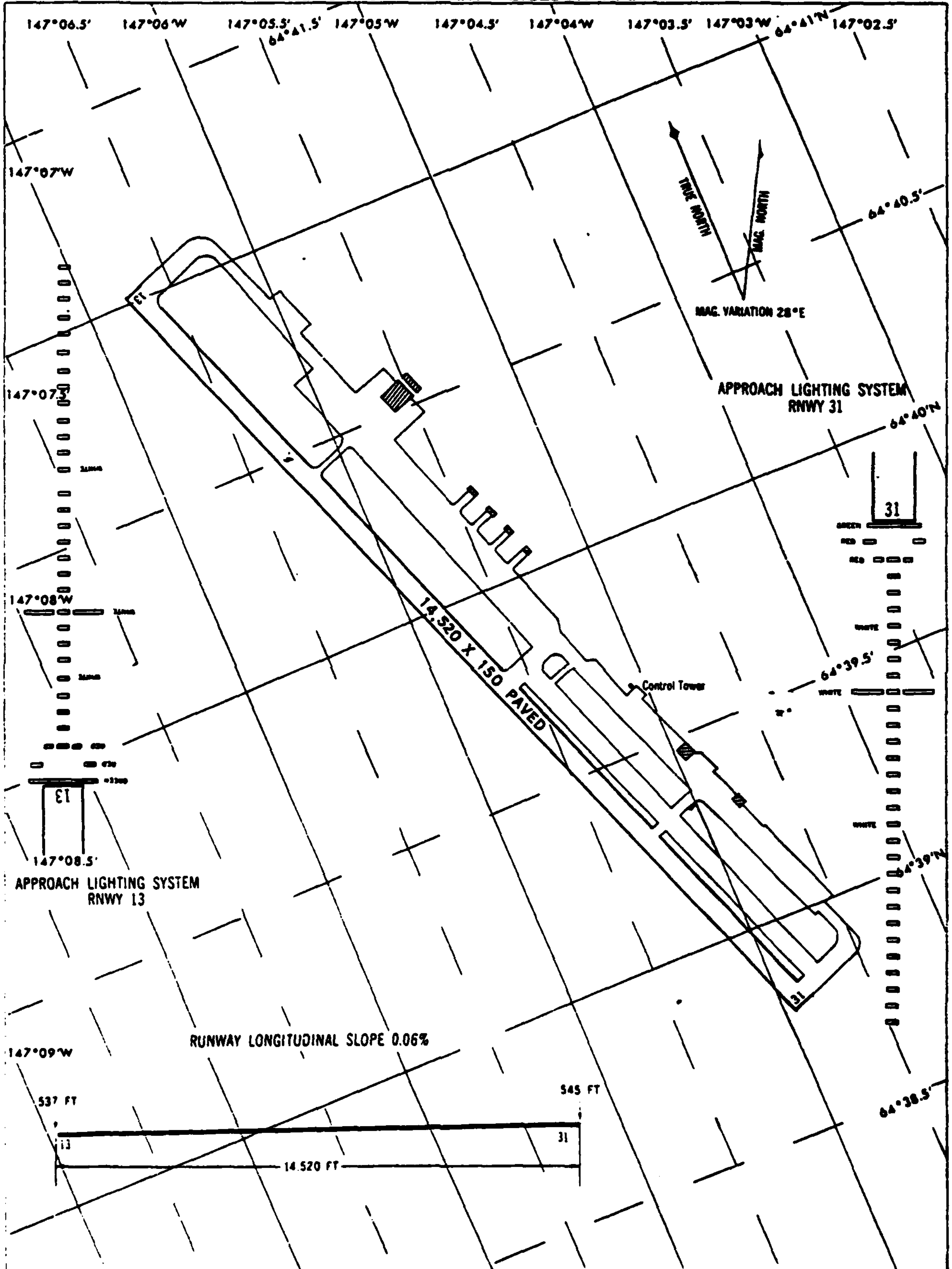


Figure 5.14. Eielson AFB

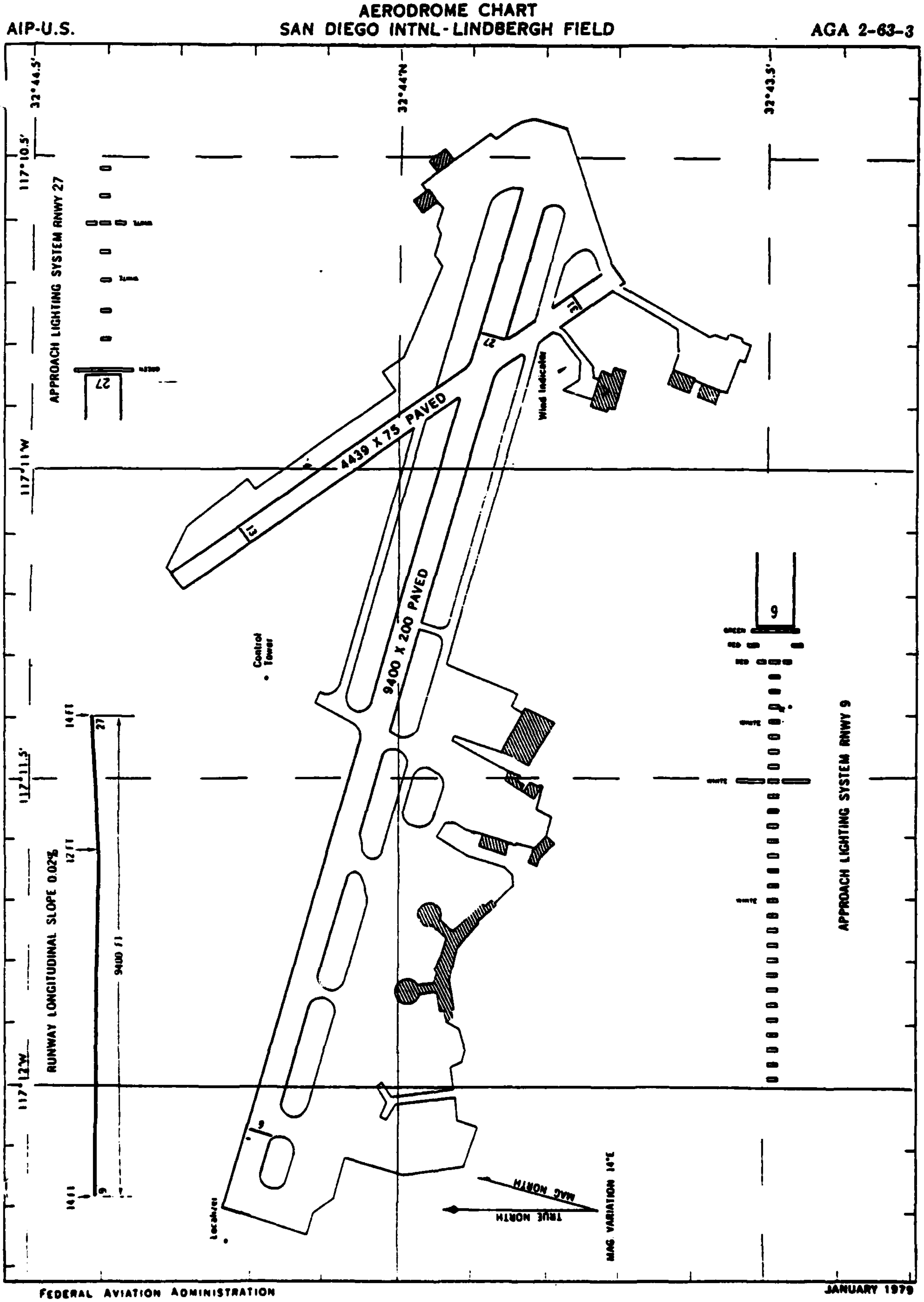


Figure 5.15. San Diego International

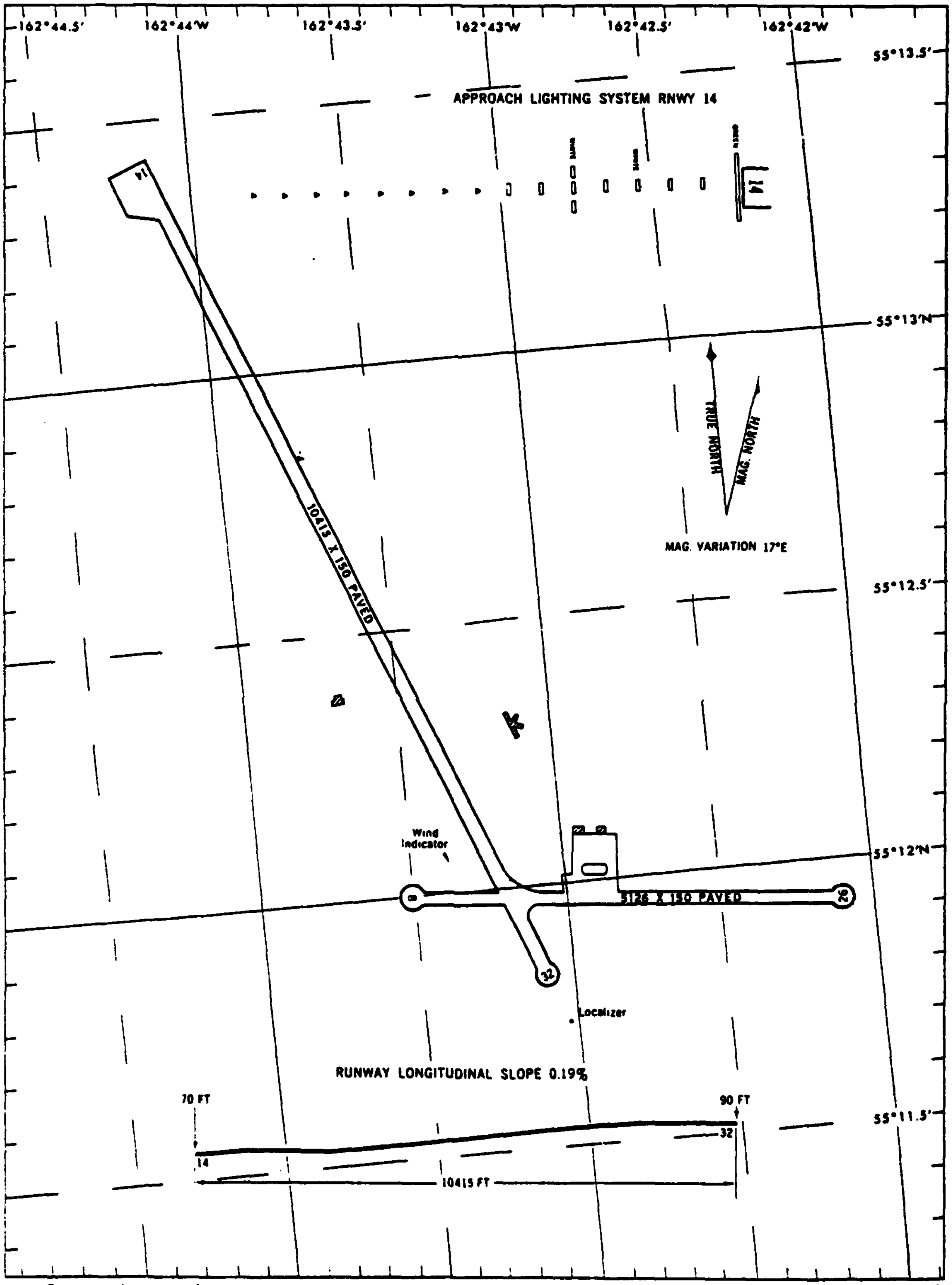


Figure 5.16. Cold Bay

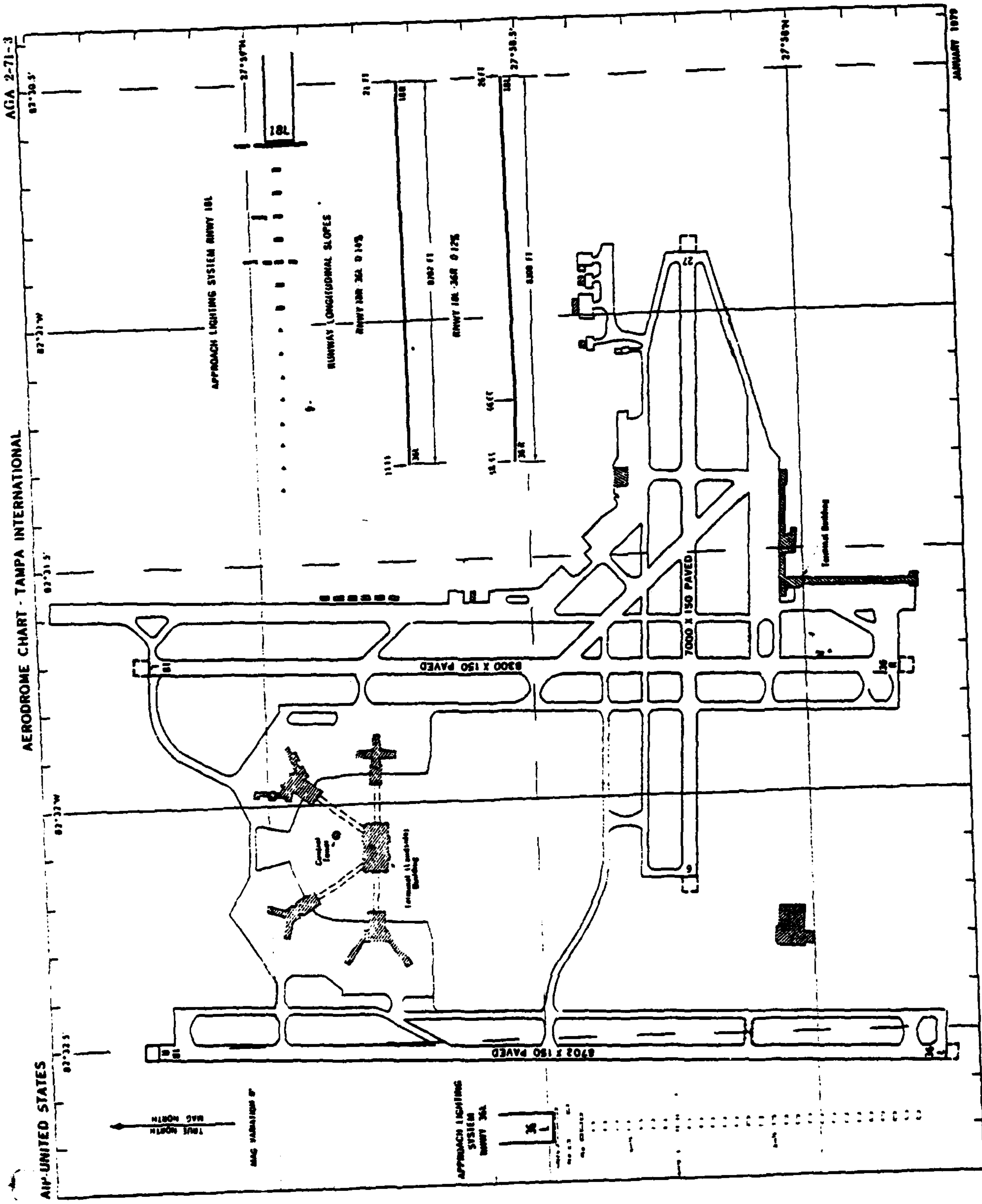


Figure 5.17. Tampa International

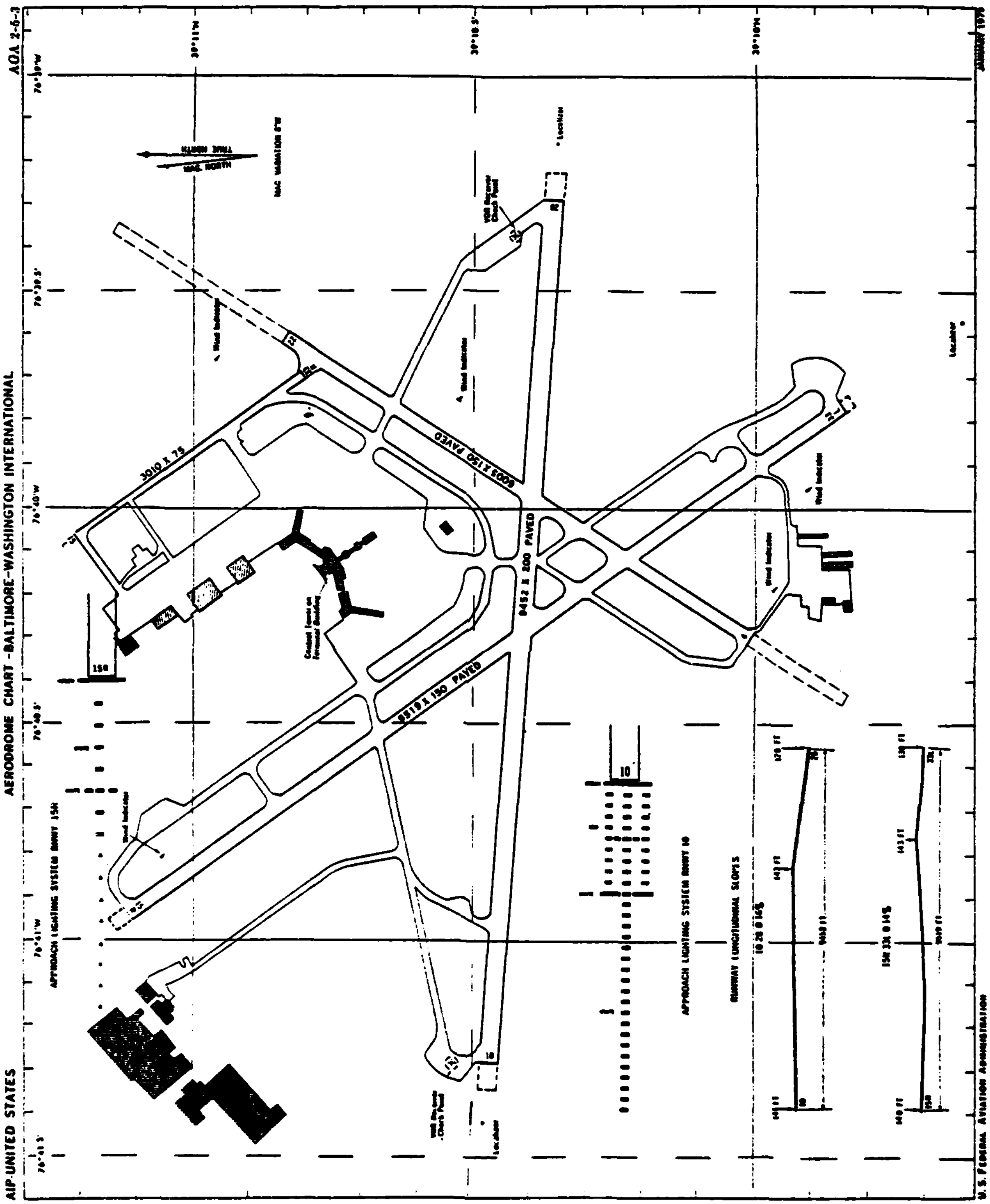


Figure 5.18. Baltimore-Washington International

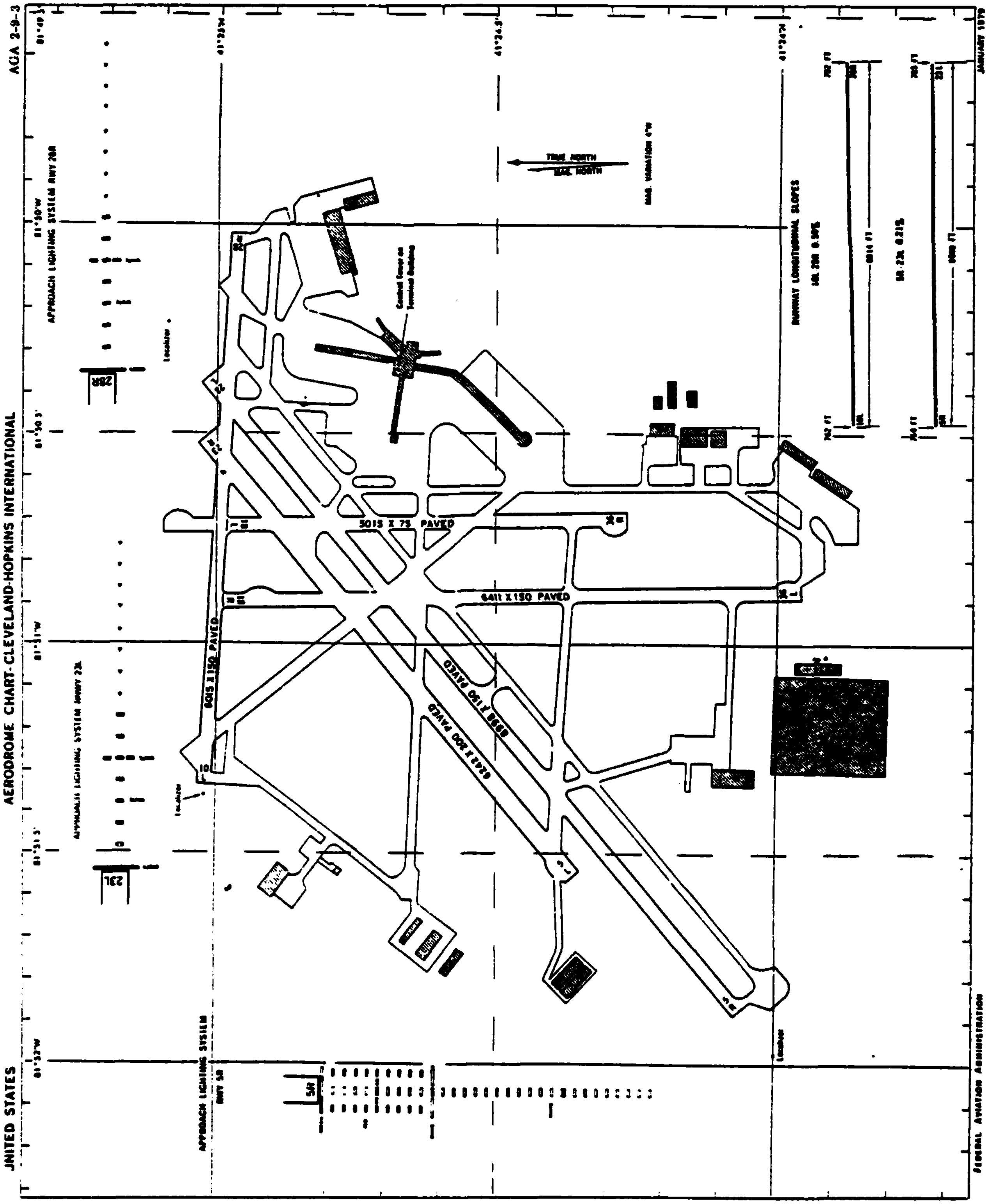


Figure 5.19. Cleveland-Hopkins International

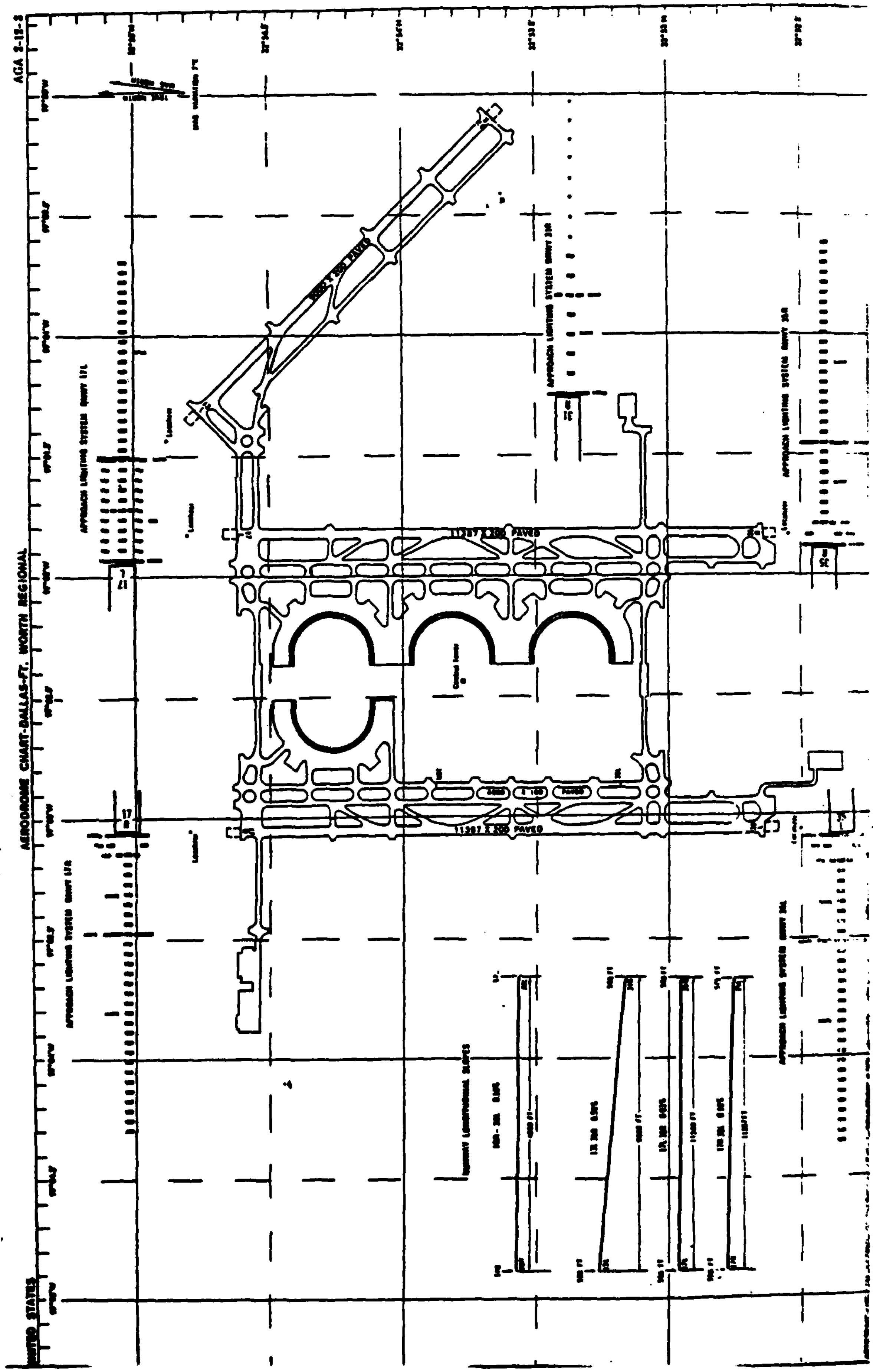


Figure 5.20. Dallas-Ft. Worth Regional

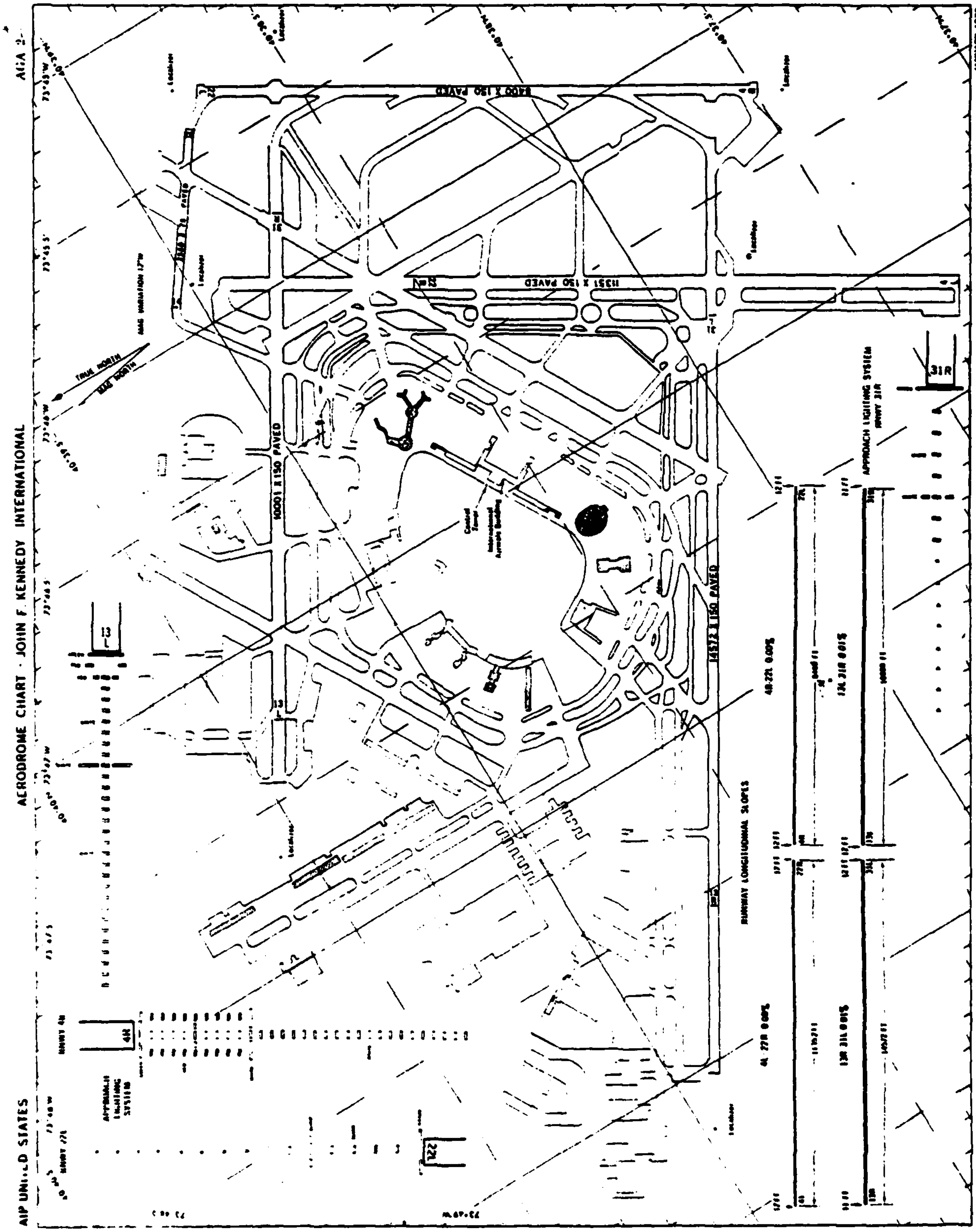


Figure 5.21. Kennedy International

descriptions for all major airports, both U.S. and international. The data in this document provide a convenient single source for characteristics like typical runway width and length, number of runways, apron geometry, and even runway slope. Although the ten presented here were selected randomly in terms of specific location, the complexity of geometry was reviewed and used as a selection device in order to obtain a wide range of shapes. The wide range of geometries displayed in the volume is perhaps one of its most interesting characteristics. The majority of the line drawings displayed in the figures are for airports located in the continental United States. These are to a large extent indicative of the images that may be constructed as the result of the application of an edge detection algorithm to the topographic images proposed for processing in the current study.

The specific airport locations depicted in the Figure are as follows:

Figure No	Location/Designation
5.12	Yap
5.13	Pago Pago International
5.14	Eielson AFB
5.15	San Diego International
5.16	Cold Bay
5.17	Tampa International
5.18	Baltimore-Washington International
5.19	Cleveland-Hopkins International
5.20	Dallas-Ft. Worth Regional
5.21	Kennedy International

The figures are arranged in a generally increasing order of geometrical complexity, starting with Yap location, which features only one long strip of moderate length (4,832 feet),

and ends with perhaps one of the most complex airport geometries in the world, John F. Kennedy Airport in New York. The latter features five operational runways ranging from 2,560 to 14,572 feet in length. The array of taxiways, aprons, and support roadways is extremely complex. These elementary components are both long and straight and curved in nature.

These figures illustrate the wide range of sizes, geometries, and complements of facilities that can be aggregated to form the functional unit that can be ascribed to the class generally called "airports." From these figures, however, several very valuable observations can be made regarding general attributes. In general, these attributes when used in conjunction with one another, form a set unique to the generic airport geometry and thus provide characteristics that can be used to uniquely identify an unknown image if that image is indeed that of an airport. The primary objective here is to identify a small set of <u>unique</u> attributes that occur often in an airport geometry and concurrently do not occur often in images.

Some of the major characteristics of the ten selected facilities are shown in Table 5.7. At the top of the table is the simplest facility, which is merely a simple single rectangular strip 100 feet in width, 4,832 feet in length, with straight ends, no taxiways, and a simple terminal facility located at approximately the 60/40 point along the strip. The facilities become increasingly complex as the entries become lower in the chart. The most complex entry, Kennedy International, New York, contrasts greatly with the first entry. As an indication of

Table 5.7

Major Characteristics of Ten Randomly Selected Airport Facilities

Facility	Number of Major Runways	Runway Lengths (ft)	Runway Widths (ft)	Total Runway Area (ft^2)	Type Geometry
Yap	1	4,832	100	483,300	Single strip
Papo-Pago	2	3,570 9,000	75 150	1,617,759	Connected-V
Eielson AFB	1	14,520	150	2,178,000	Single Strip with Parallel Taxiways
San Diego Int'l	2	4,439 9,4001	75 400	2,215,925	Crossed-V with Parallel Taxiways
Cold Bay	2	5,126 10,415	150 150	2,331,150	Simple Crossed-V
Tampa Int'l	3	7,000 8,300 8,702	150 150 150	3,600,300	Parallel Main Runways, Secondary Crossed-V All with Parallel Taxiways
Baltimore-Washington Int'l	3	6,005 9,452 9,519	150 200 150	4,745,000	3 Way Crossed-V
Cleveland-Hopkins Int'l	5	5,015 6,015 6,242 6,411 8,998	75 150 200 150 150	4,838,125	Complex
Dallas-Ft. Worth	4	4,000 9,000 11,387 11,387	100 200 200 200	6,354,800	Two Main Parallel Runways with One Additional Secondary Parallel Fourth Distant Complex Taxiway Pattern
J.F. Kennedy Int'l	5	2,560 8,400 10,001 11,351 14,572	75 150 150 150 150		Two Pairs of Main Runways at Right Angles. Very Complex Taxiway and Support Network

the degree of slope in these typical airport structures, the most level facility exhibits zero longitudinal slope over its two mile runway length and the worst exhibits a 0.5% longitudinal slope (45 feet over its 9,000 foot length). Since the largest facility, Kennedy International, exhibits the most complex structure, it serves to indicate the characteristics that can be used for identification purposes and includes many secondary characteristics that must be overlooked if the identification alogrithm is to be successful. (More specifically, the algorithm must suppress those characteristics that do not play a part in providing decisions as the decision process is carried out.)

These latter characteristics may be identified for the purposes intended here as extraneous to the identification process. If, for example, a key characteristic that cuts across all of the facilities is identified as "runway(s) with an extremely straight character and of limited length", then other characteristics of the complex target may be regarded as noise. Connecting taxiways and service roads or aprons that may connect to the main runways are examples of these extraneous attributes when viewed in the context of the single attribute identified above. This discussion and the more complex facility descriptions (Figures 5.19, 5.20, 5.21) illustrate the necessity for processing or encoding of the image or one of its transformed products with a technique that is designed to select effective features. The generic nature of the airport configuration, and the need to develop techniques that emphasize key attributes

heavily affects the selection of encoding schemes that may be considered for use in conjunction with the target recognition process. This area of interest is addressed in greater detail elsewhere in this report. Several encoding techniques, to include raster scan encoding of the raw image and Freeman encoding of the line drawing produced by an edge detection algorithm, were considered during the course of the period by looking at an empirical characterization of candidate images. The conclusion of this brief effort was that in the very complex facility-related images these techniques tend not to deemphasize the detail associated with the image complexity. The result is a very complex encoded characterization that includes the key attribute(s) of interest as well as a large amount of extraneous information mixed into the key data stream. The type of characterization forces the subsequent analysis steps to carry the burden of sorting out the key attribute data.

It is particularly important to emphasize the points elucidated in the paragraph above, since on the one hand an objective of the study was to investigate the opportunities that could be obtained from image encoding in the image identification context. On the other hand the major objective was to investigate identification schemes appropriate to the targets of interest. A general conclusion here is that in the case where the target is a complex generic target, the types of encoding schemes that were investigated in this study are of limited value. If, on the other hand, the definition of encoding is extended to include processes like the Hough transform, then encoding can

be considered a useful technique in the overall target identification process. As a matter of fact, the Hough transform technique is particularly useful in the airport identification process because its characteristics allow the emphasis of a key attribute in the airport problem--the long, straight, abruptly terminating runway and taxiway surfaces, while deemphasizing edges that do not contribute.

Let us consider the key attributes that have been identified as indicative of an airport facility regardless of the complexity and level of detail associated with the overall structure. These are identified as follows:

- One or more Runway Surfaces

- Two or more pairs of Edges

- Edges lie along extremely straight lines
- Line length 4,000 - 15,000 feet long
- Edge pairs 75-200 feet apart (150 feet appears most likely)
- Other edge pairs parallel to the first
  May be of lesser or equal width
  (Taxiways vs parallel runways)

- Most major airport facilities include paved areas of 500,000 square feet as a minimum; large scale airports contain 5,000,000 square feet of paved areas as a minimum

These very simple characteristics are present in all of the facilities identified here and certainly do not include many of the available cues identified earlier. They are, however, implemented easily and would appear to be effective features due to their unique character with respect to the other types of candidate targets expected in the images.

Implementation

The requirements for analysis and an algorithm resulting from the characteristics identified in the preceding paragraphs are very similar to those identified in the earlier bridge related analysis. Of particular importance are the requirements to identify long (4,000 - 15,000 feet) parallel lines in the images and to determine their separation distances. The Hough transform and the algorithms developed in support of the bridge identification effort will require minor alteration to allow this analysis using tools that are available. It is necessary that the edges representing the runways be exposed to a very fine degree of angle and intercept quantization to assure that the edge that is under investigation is indeed a straight edge as is characteristic of the airport runway surface. End points in the space domain must be recorded during the analysis to assure that the long parallel edges do span distances in the 4,000 to 15,000-foot range, and that the extent of the span(s) is indeed finite and ends abruptly. The inverse Hough routine described above may prove useful in carrying out the analysis related to the determination of abruptly terminating spans.

Analysis with respect to the extremely large paved surface area will be facilitated by the fact that most airport surfaces are constructed of concrete to enable them to support aircraft. These areas will be large (greater than 500,000 ft^2 and extending up through 10,000,000 ft^2) and will include the regions identified in the Hough portion of the analysis as potential runway surfaces. The extent of the large area target (bounding rectangle) in terms of coordinates will be roughly 15,000 by 15,000 feet. The analysis here will require thresholding of the raw image followed by a routine that will examine the sizes of the regions of uniformly high intensity.

CHAPTER VI

CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER WORK

Conclusions

Progress has been made during the past year in the development of computer techniques to extract items of cartographic interest from aerial photographs. The problem as originally posed was the construction of a system that can perform automatic or semi-automated cartographic analysis; a number of subsidiary problems were identified that provided the pattern-recognition framework for the overall problem. To date we have examined two types of man-made cartographic objects--bridges and runway configurations--in a variety of surroundings. Results were good, indicating that it is possible to locate accurately those objects in the image, using a combination of edge-detection and transform techniques and certain a priori information about the two types of objects.

The success to date provides support for the use and enhancement of the Hough transform as a locator of lines and line segments; since lines often characterize man-made cartographic objects, there is value in further development of that tool. The use of a large number of picture segments, coupled with a

variety of object/area types is necessary for completion evaluation of that approach to feature extraction.

The medial axis transform has been refined for use in the discrete case and is expected to be valuable in representing largearea types of objects (as opposed to bridge and runway targets).

The success of various features used in recognition of cartographic objects depends in part on the spatial resolution and intensity quantization of the digitized image. Although the present values of resolution and quantization are adequate, it may be that less of either or both would also be adequate; in cases where memory is limited one many have to make a trade between number of pixels and number of gray levels per pixel. The effect of such a trade on the effectivness of various features is unknown.

Interesting and Important Problems

It quickly becomes clear that pictures can be expressed as syntactic structures, composed of multiple occurrences of members of a small set of primitive shapes. Cartographic images can be expected to have a set of elementary components that can be combined into strings the parsing of which will yield the classes of the objects of interst. The problem is to identify such a set and demonstrate its practicality and accuracy.

The degree of invariance of features as the tradeoff (for fixed memory size) between number of pixels and number of levels occurs can be measured by a sensitivity analysis. It

will be of great theoretical and practical value to know where resources (both memory and computing) should be devoted. Further, depending on the cartographic item of interest the allocation of resources may be changed during processing. Part of this problem is affected by the goal of removing redundancy; as noted in this report, a homogeneous area of large size need not have all of its component pixels identified individually, which would allow a compact representation to guide the feature extraction.

Classifier design (except for the case of a syntactic approach) is a worthwhile area for study because it uses whatever information is available from the features to partition the space to minimize the probability or the expected cost of misclassification. One approach uses features sequentially, thus computing them only as needed to reach a preset level of error probability. Methods exist that are optimal in the sense that the number of measurements required is the minimum necessary to achieve the specified error.

It is essential to extend the work already done to new types of objects, to larger numbers of them (to obtain good estimates of classifier performance), to a variety of backgrounds and qualities of photographs, and to imagery of other kinds (radar, infrared, etc.).

REFERENCES

[1] I. Abdou and W. K. Pratt, "Quantitative Design and Evaluation of Enhancement/Thresholding Edge Detectors," Proc. IEEE, Vol. 67, No. 5, 753-763, 1979.

[2] H. Blum, "A Transformation for Extracting New Descriptors of Shape," In Waltham-Dunn, Weiant, Eds., Symposium on Models for the Perception of Speech and Visual Form, Cambridge, MA., MIT Press, 1964.

[3] L. Calabi, "On the Shape of Plane Figures," PML No. 60249,. SR-1, March 1965.

[4] L. Calabi, "A Study of the Skeleton of Plane Figures," PML No. 60429, SR-2, June 1965.

[5] L. Calabi, "The Skeletal Pair Determines the Convex Deficiency," PML No. 5711 TM-1, January 1966.

[6] L. Calabi, "A Study of the Skeletal Graph," AFCRL-68-0498, August 1968.

[7] L. Calabi, "The Many Faces of the Skeleton," AFCRL-69-0405, September 1969.

[8] L. Calabi and W. E. Hartnett, "Shape Recognition, Prairie Fires, Convex Deficiencies, and Skeletons," American Mathematics Monthly, 75, 1968, pp. 335-342.

[9] K. Castleman, Digital Image Processing, Englewood Cliffs, NJ, Prentice-Hall, 1979.

[10] E. S. Deutsch and J. R. Fram, "A Quantitative Study of the Orientation Bias of Some Edge Detector Schemes," IEEE Trans. Comput., Vol. C-27, No. 3, 205-213, March 1978.

[11] R. O. Duda and P. E. Hart, Pattern Classification and Scene Analysis, New York, NY, J. Wiley, 1973.

[12] R. O. Duda and P. E. Hart, "Use of the Hough Transform to Detect Lines and Curves in Pictures," Commun. of the ACM, Vol. 15, No. 1, 11-15, January 1972.

[13] J. R. Fram and E. S. Deutsch, "On the Evaluation of Edge Detector Schemes and Their Comparison with Human Performance," IEEE Trans. Comput., Vol. C-24, No. 6. 616-628, June 1975.

[14] W. Frei and C. C. Chen, "Fast Boundary Detection: A Generalization and New Algorithm," IEEE Trans. Comput., Oct. 1977.

[15] F. Gorman and M. B. Clowes, "Finding Picture Edges Through Collinearity of Feature Points," IEEE Trans. Comput., Vol. C-25, No. 4, 449-456, April 1976.

[16] R. M. Haralick, "Automatic Remote Sensor Image Processing," in Topics in Applied Physics: Digital Picture Analysis, A. Rosenfeld, Ed., Springer-Verlag, NY, 5-58, 1976.

[17] R. M. Haralick, "Statistical and Structual Approaches to Texture," Proc. IEEE, Vol. 67, No. 5, 786-804, 1979.

[18] J. K. Hawkins, "Textural Properties for Pattern Recognition," in Picture Processing and Psychopictorics, Lipkin and Rosenfeld, Eds., Academic Press, Inc., 1969.

[19] J. Hilditch, "An Application of Graph Theory in Pattern Recognition," in Machine Intelligence, Vol. 3, D. Michie, Ed., Edinburgh University Press, Edinburg, Scotland, 1969

[20] J. Hilditch, "Linear Skeletons from Square Cupboards," in Machine Intelligence, Vol. 4, B. Meltzer and D. Michie, Eds., Edinburg University Press, Edinburg, Scotland, 1969.

[21] R. Kirsch, "Computer Determination of the Constituent Structures of Biological Images," Comput. Biomed. Res., Vol. 4, No. 3, 315-328, 1971.

[22] J. Kotelly, "A Mathematical Model of Blum's Theory of Pattern Recognition," AFCRL-63-164, April 1963.

[23] G. Levi and U. Montanari, "A Gray-weighted Skeleton," Information and Control, 17, 1970, pp. 62-91.

[24] U. Montanari, "A Method for Obtaining Skeletons Using A Quasi-Euclidean Distance," J. ACM, 15, October 1968, pp. 600-624.

[25] U. Montanari, "Continuous Skeletons from Digitized Images," J. ACM, 16, October 1969, pp. 534-549.

[26] J. C. Mott-Smith, "Medial Axis Transformations," in Picture Processing and Psychopictorics, B. S. Lipkin and A. Rosenfeld, Eds., New York, NY, Academic Press, 1970.

[27] R. Nevatia, "Locating Object Boundaries in Textured Environments," IEEE Trans Comput., 1170-1175, November 1976.

[28] T. Pavlidi, "A Review of Algorithms for Shape Analysis," Computer Graphics and Image Processing, 7, 1978, pp. 243-258

[29] O. Philbrick, "Shape Description with the Medial Axis Transformation," in Pictorial Pattern Recognition, Washington, DC, Thompson Book Company, 1968.

[30] W. K. Pratt, Digital Image Processing, New York, NY, J. Wiley & Sons, 1978.

[31] J.M.S. Prewitt, "Object Enhancement and Extraction," in Picture Processing and Psychopictorics, Lipkin and Rosenfeld, Eds., Academic Press, NY, 1970.

[32] J. Riley, "Plane Graphs and Their Skeletons," PML No. 60429, TM-1, March 1965.

[33] L. G. Roberts, "Machine Perception of 3-dimensional Solids," in Optical and Electro-optical Information Processing, J. T. Tippet, et. al., Eds., MIT Press, Cambridge, MA, 159-197, 1965.

[34] A. Rosenfeld and J. L. Pfaltz, "Sequential Operations in Digitized Picture Processing," J. ACM, 13, October 1966, pp. 471-494.

[35] A. Rosenfeld and M. Thurston, "Edge and Curve Detection for Scene Analysis," IEEE Trans. Comput., Vol. C-20, 562-569, May 1971.

[36] C. Dyer and A. Rosenfeld, "Thinning Algorithms for Grayscale Pictures," University of Maryland, 1977.

[37] W. B. Thompson, "Textural Boundary Analysis," IEEE Trans. Comput., March, 1977, 574-577.

[38] R. J. Wall, A. Klinger, and S. Harami, "An Algorithm for Computing the Medial Axis Transform and Its Inverse," Proceedings of the 1977 Workshop on Picture Data Description and Management, pp. 121-122, 77CH1187-4C Piscataway, NJ, IEEE, 1977.

[39] H. Andrews, "Bibliography on Rate Distortion Theory," IEEE Transactions on Information Theory, Vol. IT -17, No. 2, March 1971.

[40] J. R. Bennett and J. S. MacDonald, "On the Measurement of Curvature in a Quantized Environment," IEEE Transactions on Computers, Vol. C-24, No. 8, August 1975, pp. 803-820.

[41] L. Davisson, "Rate-Distortion Theory and Applications," Proceedings of the IEEE, Vol. 60, No. 7, July 1972, pp 800-808.

[42] W. H. Foy, Jr., "Entropy of Simple Line Drawings," IEEE Transactions on Information Theory, Vol. IT-10, No. 4, April 1964, pp. 165-167.

[43] H. Freeman, "On the Encoding of Arbitrary Geometric Configurations," IRE Transactions on Electronic Computers, Vol. EC-10, No. 2, June 1961, pp. 260-268.

[44] H. Freeman and J. Glass, "On the Quantization of Line-Drawing Data," IEEE Transactions on Systems, Science and Cybernetics, Vol. SSC-5, No. 1, January 1969, pp. 70-72.

[45] H. Freeman, "Computer Processing of Line-Drawing Images," Computing Surveys, Vol. 6, 1974, pp. 57-97.

[46] R. G. Gallager, Information Theory and Relilable Communication, J. Wiley & Sons, Inc., New York, NY, 1968.

[47] R. C. Gonzalez and P. Wintz, Digital Image Processing, Additon-Wesley Publishing Co., Inc., Reading, MA, 1977.

[48] D.N. Graham, "Image Transmission by Two-Dimensional Contour Coding," Proceedings of the IEEE, Vol. 55, No. 3, March 1967, pp 336-346.

[49] D. Sakrison and V. R. Algazi, "Comparison of Line-by-Line and Two-Dimensional Encoding of Random Images," IEEE Transactions on Information Theory, Vol. IT-17, No. 4, July 1971, pp. 386-398.

[50] L. Shapiro, "Data Structures for Picture Processing: A Survey," Computer Graphics and Image Processing, 1977, pp. 162-184.

[51] M. Tasto and P. A. Wintz, "A Bound on the Rate Distortion Function and Application to Images," IEEE Transactions on Information Theory, Vol. IT-18, No. 1, January 1972, pp. 150-159.

```
      [illegible]
      [illegible]
      K=LH/(N-1)
      INDEX=0
      DISPLAY "COLLAPSE(0) OR DOWNSAMPLE(1)"
      ACCEPT IPROC
      IF(IPROC.EQ.1) GO TO 10
    1 DO 2 J=1,N
      IC(J)=K**2
    2 IB(J)=0
      DO 4 J=1,K
      READ(IIF,END=20) (IA(M),M=1,LH)
      DO 4 I=1,LH
      IJK=1+(I-1)/K
      IF(IA(I).GE.0.AND.IA(I).LE.255) GO TO 3
      IC(IJK)=IC(IJK)-1
      GO TO 4
    3 IB(IJK)=IB(IJK)+IA(I)
    4 CONTINUE
      DO 5 I=1,N-1
      IF(IC(I).EQ.0) IC(I)=1
    5 IB(I)=IB(I)/IC(I)
      WRITE(IOF,END=20) (IB(M),M=1,N-1)
      INDEX=INDEX+1
      GO TO 1
   10 DO 15 J=1,LH,K
      READ(IIF@J,END=20) (IA(M),M=1,LH)
      DO 13 I=1,LH,K
      IJK=I/K +1
   13 IB(IJK)=IA(I)
      WRITE(IOF,END=20) (IB(M),M=1,N-1)
   15 INDEX=INDEX+1
   20 REWIND IIF
      REWIND IOF
      IJK=N-1
      DISPLAY "OLD SIZE=",LH," NEW SIZE=",IJK," BY",INDEX
      RETURN
      END
```

Appendix 3.1. Downsampling Algorithm

```
[illegible] (LH,N,IX,IY,I[illegible]
C-[illegible] THER NOISE SMO[illegible]
E
      C-SPIKES ON A SUBIMAGE(OR IMAGE)OF SIZE N LOCATED IN FILE IIN.
      C-IF SPIKE ELIM IS SELECTED(NTH=4) EACH PIXEL IS COMPARED TO
      C-TO THE MEAN OF ITS 8 NEIGHBORS A SET EQUAL TO THE MEAN IF IT
S
      C-VALUE IS MORE THAN A SPECIF # OF STD DEV'S(AX) AWAY FROM IT.
      C-FOR NOISE SMOOTHING, 1 OF 3 CONVOL. MASKS IS USED. YF NTH=1,
      C-THE MASK HAS ALL ENTRIES=1. YF NTH=2, THE CENTRAL PIXEL IS
      C-DOUBLE WEIGHTED.IF NTH=3, THE CORNERS=1,SIDE-SHARING PIXELS=
2
      C-AND THE CENTRAL PIXEL IS WEIGHTED BY 4. LH IS THE INPUT REC'
D
      C-SIZE,IX & IY THE STARTING COORD'S, AND IND THE FILE INDEX PA
RAMETER
            COMMON R1(1000),R2(1000),R3(1000)
            INTEGER R1,R2,R3,BR(N)
            IF(IX.LE.1) IX=2
            IF(IY.LE.1) IY=2
          1 INDEX=(IND-1)*N
            READ(IIN@(IX-1)) (R1(I),I=1,LH)
            READ(IIN@IX) (R1(I),I=1,LH)
            RTY=NTH
            DO 19 K=IX+1,IX+N
            READ(IIN@K,END=20) (R3(I),I=1,LH)
            GO TO (4,4,8,10),NTH
          4 DO 6 J=IY,IY+N-1
          6 BR(J-IY+1)=(8.*PS(J,1)+RTY*R2(J))/(8.+RTY)
            GO TO 15
          8 DO 9 J=IY,IY+N-1
          9 BR(J-IY+1)=(R1(J-1)+R2(J+1)+R3(J-1)+R3(J+1)+4.*R2(J)
           X +2.*(R1(J)+R3(J)+R2(J-1)+R2(J+1)))/16.
            GO TO 15
         10 DO 12 J=IY,IY+N-1
            XX=PS(J,1)-R2(J)
            XY=(8./7.)*(PS(J,2)-(XX+R2(J))**2
            XZ=SQRT(XY)
            BR(J-IY+1)=R2(J)
            IF(ABS(XX).GT.(AX*XZ)) BR(J-IY+1)=XX+R2(J)
         12 CONTINUE
         15 WRITE(IO1@(INDEX+K-IX)) (BR(I),I=1,N)
            DO 16 J=IY-1,IY+N
            R1(J)=R2(J)
         16 R2(J)=R3(J)
         19 CONTINUE
         20 REWIND IIN
            RE[illegible]
            RETURN
            END
```

Appendix 3.2. Algorithm for Noise-Cleaning

```
      PROGRAM HOUGHTRAN
      IMPLICIT INTEGER*2 (I,J,K,L,M,N)
-THIS SECTION REQUESTED THE PARAMETERS NEEDED BY THE VARIOUS
-ROUTINES OF THE PROGRAM.
      CHARACTER*1 IAN,NTY,IAN2
      CHARACTER*5 FNTWO
      CHARACTER*9 FNAME
-THE NEXT 2 LINES ASK IF THE USER WISHES TO PROCESS INTER-
-ACTIVELY (ON 1 SUBIMAGE AT A TIME) OR AUTOMATICALLY (PROCESS
-ENTIRE IMAGE WITHOUT USER INTERACTION).
    1 WRITE(6,*)'PROCESSING MODE (0=INTERACTIVE, 1=AUTOMATIC)'
      READ(5,*) MP
      WRITE(6,*) 'QUANT.LEVELS OF RHO(<256)& THETA(<13)'
      READ(5,*) LR,LA
      WRITE(6,*) '# OF ROWS(<256) AND COLS (<256) TO PROCESS'
      READ(5,*) NX,NY
      WRITE(6,*)'WINDOW SIZE FOR GLOBCON TEST (DEFAULT=7):'
      NW=7
      READ(5,*) NW
      NW=(NW-1)/2
      IF(MP.EQ.1) GO TO 2
-THE FOLLOWING ASKS WHAT OUTPUT TO PERFORM DURING ROUTINE
      WRITE(6,*) 'DISPLAY EDGE FIELD? (Y/N)'
-IF 'Y' IS ENTERED, THE RESULTS OF EDGE DETECTION IS OUTPUT
      READ(5,500) NTY
  500 FORMAT(A1)
      WRITE(6,*) 'DISPLAY HOUGH TRANSFORM MATRIX? (Y/N)'
-IF 'Y', THE TRANSFORM MATRIX IS OUTPUT
      READ(5,500) IAN2
      WRITE(6,*) 'DISPLAY INVERTED TRANSFORM? (Y/N)'
-IF'Y', INVERTED TX DISPLAYED SUPERIMPOSED ON EDGE FIELD
      READ(5,500) IAN
-THE FOLLOWING REQUESTS THRESHOLDS FOR VARIOUS TESTS INVOLVED
-IN EXTRACTING LINES, BRIDGE CANDIDATES, AND BRIDGES
    2 WRITE(6,*)'FOR FOLLOWING THRESHOLDS, ENTER 0 FOR DEFAULT'
      WRITE(6,*) 'EDGE DETECTOR THRESHOLD (DEFAULT=5000.0):'
      READ(5,*) AX
      IF(AX.EQ.0) AX=5000.0
      WRITE(6,*) 'MATRIX INVERSION THRESHOLD (DEFAULT=10):'
      READ(5,*) IG
      IF(IG.EQ.0) IG=10
      WRITE(6,*)'PARA.LINE SEPAR. THRESHOLD(DEFAULT=5 PIXELS)'
      READ(5,*) ST
      IF(ST.EQ.0) ST=5.0
      WRITE(6,*)'MAX.LINE PAIR OFFSET (% OF AVG LNTH,DEFAULT=.5)'
      READ(5,*) OT
      IF(OT.EQ.0) OT=0.5
      WRITE(6,*)'MAX LENGTH DIFFERENCE(% OF AVG LGNTH)(DEFAULT=0.5):'
      READ(5,*) DT
      IF(DT.EQ.0) DT=0.5
      WRITE(6,*)'UNIFORMITY THRESHOLD FOR GLOBCON(DEFAULT=15.0):'
      READ(5,*) BT
      IF(BT.EQ.0) BT=15.0
```

Appendix 3.3. Bridge-Detection Program

```
  3 WRITE(6,*) 'INPUT FILE NAME (5 DIGITS; IE, PIC05)'
    READ(5,501) FNTWO
501 FORMAT(A5)
    FNAME=FNTWO//'.DAT'
    WRITE(6,*) 'INPUT RECORD LENGTH'
    READ(5,*) LH
    IF(MP.EQ.1) GO TO 20
    WRITE(6,*)'STARTING COORDS(UPPER-LEFT CORNER, ROW,COL):'
    READ(5,*) IX,IY
-IMAGE TO BE OPERATED ON IS READ INTO MEMORY
    CALL RIM(NX,NY,IX,IY,FNAME,LH,BR)
-HOUGH TRANSFORM IS PERFORMED
    CALL HOUGH(LR,LA,IX,IY,NX,NY,AX,NTY)
-MATRIX IS OPTIONALLY DISPLAYED
    IF(IAN2.EQ.'N') GO TO 4
    CALL MATRIX(LR,LA)
-THE TRANSFORM IS INVERTED (IE, LINES ARE EXTRACTED)
  4 CALL INVERT(LR,LA,IG,IX,IY,NX,NY,IAN,N)
-CANDIDATES ARE EXTRACTED AND TESTED USING GLOBCON
    CALL PARLINES(N,IX,IY,ST,OT,DT,BT,FNAME,LH,'Y',0,NW)
    GO TO 5
-THE FOLLOWING SECTION PROCESSES AN ENTIRE IMAGE WITHOUT
-FURTHER INTERACTION BY CALCULATING EACH SUCCEEDING SET
-OF STARTING COORD'S AND USING THE VALUES ENTERED ABOVE.
 20 WRITE(6,13)
 13 FORMAT('0','CTR(X,Y)  ANGLE LNGTH WIDTH #LINES RESULT')
    DO 15 K=2,1024,NX
    NY2=NY
    NX2=NX
    DO 15 J=2,LH-2,NY
    IF((LH-J).LT.NY) NY2=LH-J
    CALL RIM(NX2,NY2,K,J,FNAME,LH,BR)
    IF(NX2.EQ.0) GO TO 5
    CALL HOUGH(LR,LA,IX,IY,NX2,NY2,AX,'N')
    CALL INVERT(LR,LA,IG,IX,IY,NX2,NY2,'N',N)
 15 CALL PARLINES(N,K,J,ST,OT,DT,BT,FNAME,LH,'Y',1,NW)
  5 WRITE(6,*)'NEXT PROCEDURE: ENTER 1-DIGIT CODE'
    WRITE(6,*) '    1) PROCESS ANOTHER IMAGE-NO CHANGES'
    WRITE(6,*) '    2) PROCESS ANOTHER IMAGE-NEW THRESHOLDS'
    WRITE(6,*) '    3) PROCESS ANOTHER IMAGE-NEW OPTIONS'
    WRITE(6,*) '    4) STOP'
-IF '1', ONLY IMAGE FILE/COORDS ARE REQUESTED. IF'2', NEW
-THRESHOLD VALUES ARE REQUESTED ALSO. '3' RESTARTS PROGRAM.
    READ(5,*) NPROC
    GO TO (3,2,1,99),NPROC
 99 STOP
    END
    SUBROUTINE HOUGH(LR,LA,IX,IY,NX,NY,AX,NTY)
    IMPLICIT INTEGER*2 (I,J,K,L,M,N)
-THIS ROUTINE PERFORMS A 'FAST' HOUGH TRANSFORM (DUE TO
-O'GORMAN & CLOWES) ON THE IMAGE ARRAY 'R'. FEATURE POINTS
-TO TRANSFORM ARE GENERATED BY A 3X3 EDGE DETECTER MASK.
-THE TRANSFORM IS 'FAST' SINCE ONLY A SINGLE RHO/THETA CELL
-IS INCREMENTED PER FEATURE POINT--THAT ONE CORRESPONDING TO
-THE ANGLE RETURNED BY THE EDGE DETECTER.
```

```
      COMMON R(256,256)/HUFF/HF(13,256,4),NTOT(256),XX
      COMMON /CH/IMG(256,256)/LT/A(13,3),PI
      INTEGER*2 HF,R,RHO
      CHARACTER*1 IMG,NTY
      PI=ATAN(1.0)*4.
-INITIALIZE THE RHO/THETA MATRIX:
      DO 1 K=1,LR
      DO 1 J=1,LA
      DO 1 L=1,4
    1 HF(J,K,L)=0
      IF(NTY.NE.'Y') GO TO 99
      WRITE(6,98) IX,IY,AX
   98 FORMAT('0','EDGE FIELD, COORDS',2I5,' THRESHOLD=',F8.1)
-XX, CALCULATED IN THE NEXT STEP, CONVERTS RHO (DISTANCE)
-TO 'RHO', THE BUCKET NUMBER.
   99 XX=(LR-1.)/(NX+NY)
-THE NEXT 10 LINES DEVELOP A 'LOOK-UP' TABLE TO FACILITATE
-THE CALCULATION OF THE RHO/THETA BUCKET FOR EACH POINT.
      DO 4 K=1,LA
      A(K,1)=(K-1)*PI/LA
      A(K,2)=A(K,1)
      A(K,3)=1
      IF(A(K,2).LE.(PI/4.).OR.A(K,2).GE.(.75*PI)) GO TO 3
      A(K,2)=(PI/2.)-A(K,2)
      IF(COS(A(K,1)).LT.0) A(K,3)=NX*XX+1
      GO TO 4
    3 IF(COS(A(K,1)).LT.0) A(K,3)=NY*XX+1
    4 CONTINUE
-THE REMAINDER PERFORMS THE TRANSFORM. FOR EACH PIXEL, THE
-EDGE OPERATOR 'EDGOP' IS CALLED, WHICH RETURNS AN EDGE
-INTENSITY B2, ANGLE A2, AND A THIRD VALUE 'IS' WHICH IS
-+1 IF A2 IS BETWEEN 0 & PI, AND -1 IF A2 IS BETW. PI & 2PI.
-B2=0 IF THE PIXEL IS NOT AN EDGE (IE, FEATURE) POINT. THE
-CORRESP. L/RHO BUCKET IS THEN CALCULATED. LINES ARE HANDLED
-SLIGHTLY DIFFERENT DEPENDING ON ITS ANGLE: THOSE BETWEEN
-PI/4 AND 3PI/4 CALCULATE RHO AS A HORIZ. AXIS INTERCEPT.
-THOSE < PI/4 OR >3PI/4 USE THE VERT. AXIS INTERCEPT AS RHO.
-LINES WITH ANGLES >PI/2 (NEG. SLOPES) MUST HAVE EITHER NX OR
-NY ADDED TO RHO IN ORDER THAT RHO 'BUCKET NUMBER' NEVER BE
-NEGATIVE. THIS PROCEDURE DIFFERS FROM THE MORE COMMON USE
-OF RHO AS THE PERPENDICULAR DISTANCE OF ANY LINE TO THE
-ORIGIN, BUT HAS THE ADVANTAGE OF NOT GIVING SPURIOUS 'SPLIT'
-LINES DUE TO IMAGE DIGITIZATION.
      DO 5 K=1,NX
      DO 5 J=1,NY
    5 IMG(K,J)=' '
      DO 15 K=1,NX
      DO 10 J=1,NY
      CALL EDGOP(J+1,K+1,AX,B2,A2,IS)
      IF(B2.EQ.0) GO TO 10
      IMG(K,J)='*'
      L=A2*LA/PI+1
```

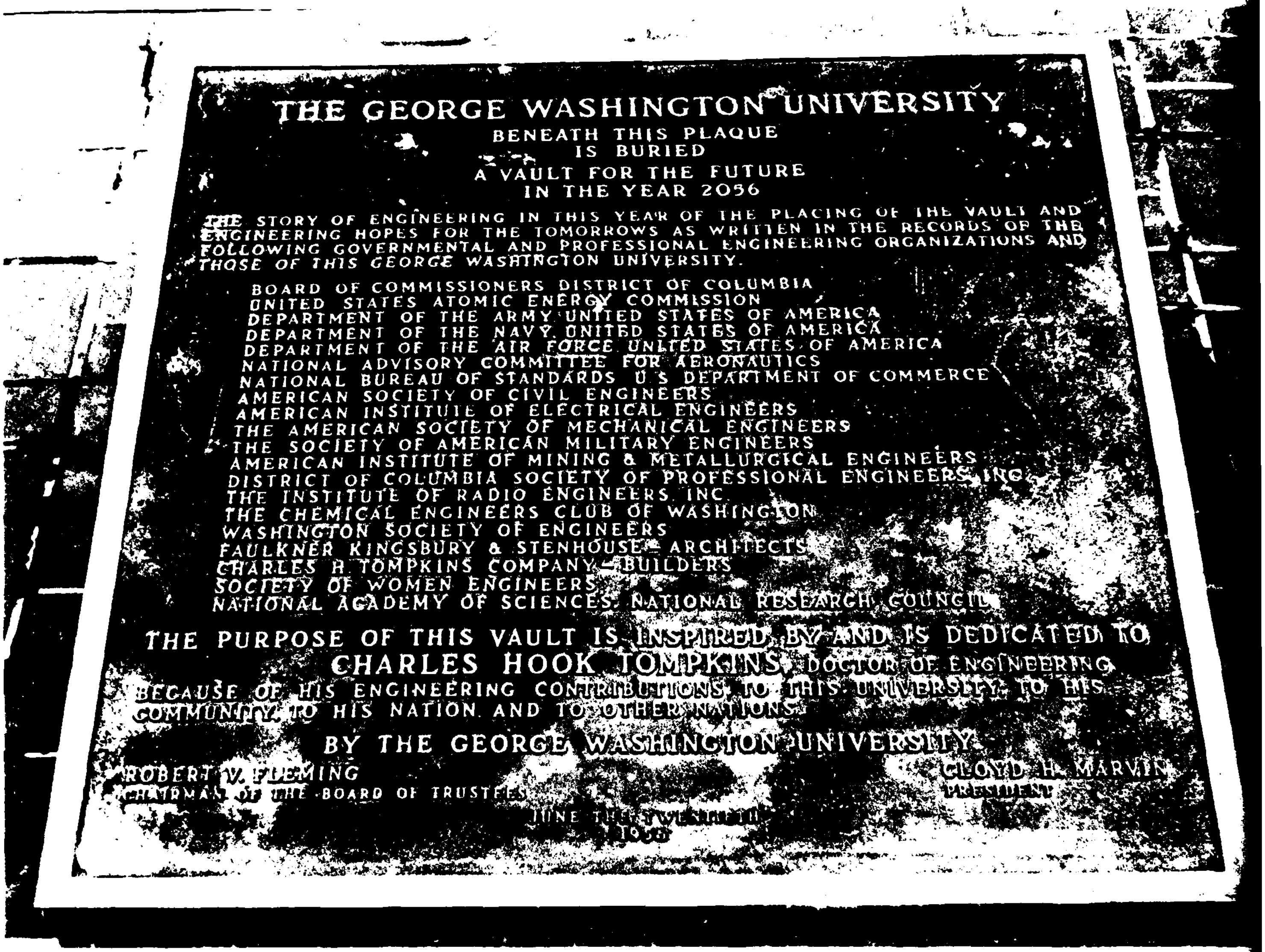
```
-BESIDES THE USUAL INCREMENTING, EACH RHO/THETA CELL KEEPS 2
-OTHER VALUES: THE AVERAGE X VALUE (A2 >3PI/4 OR <PI/4) OR
-Y VALUE (PI/2<A2<3PI/4) FOR THE 1ST 3 PTS AND FOR ALL PTS
-ALONG THAT LINE;(THESE ARE USED LATER TO ESTIMATE ITS CENTER
-AND LENGTH); THE 3RD ADD. VALUE SUMS THE 'IS' VALUES, THUS
-INDICATING IF THE LINE IS PRIMARILY 'POSITIVE'(0-PI) OR
-'NEGATIVE'(PI-2*PI).
      RHO=(J*SIN(A(L,1))+K*COS(A(L,1)))*XX/COS(A(L,2))+A(L,3)
      IV=J
      IF(HF(L,RHO,3).GT.31700) GO TO 10
      HF(L,RHO,1)=HF(L,RHO,1)+1
      IF(A(L,1).GT.(PI/4.).AND.A(L,1).LT.(.75*PI)) IV=K
      HF(L,RHO,3)=HF(L,RHO,3)+IV
      IF(HF(L,RHO,1).LE.3) HF(L,RHO,2)=HF(L,RHO,2)+IV
      HF(L,RHO,4)=HF(L,RHO,4)+IS
   10 CONTINUE
-DISPLAYS THE JUST-PAST LINE OF EDGE POINTS IF NTY='Y'.
      IF(NTY.EQ.'Y') WRITE(6,*) (IMG(K,J),J=1,NY)
   15 CONTINUE
      RETURN
      END
      SUBROUTINE EDGOP(J,K,AX,B2,A2,IS)
      IMPLICIT INTEGER*2 (I,J,K,L,M,N)
-THIS ROUTINE IS A MODIFICATION OF THE 'SOBEL' EDGE OPERATOR.
-IT USES THE WEIGHT SQRT(2) (DUE TO FREI & CHEN) RATHER THAN
-2-RESULTING IN UNIFORM PROB OF EDGE DETECTION FOR ANY ANGLE.
      COMMON R(256,256)/LT/A(13,3),PI
      INTEGER*2 R
      E=R(K-1,J-1)-R(K+1,J+1)
      D=R(K-1,J+1)-R(K+1,J-1)
      F=E+D+1.414*(R(K-1,J)-R(K+1,J))
      G=E-D+1.414*(R(K,J-1)-R(K,J+1))
      B2=F**2+G**2
      IF(B2.LT.AX) B2=0
      IS=1
-'IS' IS SET TO -1 IF THE ANGLE IS BETWEEN PI AND 2*PI.
      IF(G.LT.0) IS=-1
      IF(F.EQ.0) GO TO 2
      A2=ATAN2(G,F)
-THE NEXT LINE SHIFTS RANGE FROM (-PI/2,PI/2) TO (0,PI)
      IF(A2.LT.0) A2=PI+A2
      GO TO 3
    2 A2=(PI/2.)
    3 RETURN
      END
      SUBROUTINE MATRIX(LR,LA)
      IMPLICIT INTEGER*2 (I,J,K,L,M,N)
-THIS ROUTINE DISPLAYS THE RHO/THETA 'INCREMENT' CELLS.
      COMMON /HUFF/HF(13,256,4),NTOT(256),XX
      CHARACTER*3 DASH
      INTEGER*2 HF
      WRITE(6,1) (J,J=1,LA)
    1 FORMAT(' ',8X,20I3)
      DASH='---'
```

```
      WRITE(6,2) (DASH,J=1,LA)
    2 FORMAT(' ',9X,20A3)
      DO 5 K=1,LR
    5 WRITE(6,6) K,(HF(L,K,1),L=1,LA)
    6 FORMAT(' ',I4,4X,20I3)
      RETURN
      END
      SUBROUTINE RIM(NX,NY,IX,IY,FNAME,LH,BR)
      IMPLICIT INTEGER*2 (I,J,K,L,M,N)
-THIS ROUTINE READS IN AN IMAGE OF LENGTH LH, LOCATED IN
-FILE IN. 'NX' ROWS BY 'NY' COLUMNS (STARTING WITH UPPER
-LEFT PIXEL (IX,IY) ARE STORED IN THE INTEGER*2 ARRAY 'R'.
      COMMON R(256,256)
      INTEGER*2 R,BR(LH)
      CHARACTER*9 FNAME
      OPEN(ACCESS='DIRECT',NAME=FNAME,TYPE='OLD', UNIT=21)
      DO 5 K=IX-1,IX+NX
      READ(21'K,END=10) BR
      DO 5 J=IY-1,IY+NY
    5 R(K-IX+2,J-IY+2)=BR(J)
      GO TO 20
   10 NX=K-IX+1
   20 CLOSE(UNIT=21)
      RETURN
      END
      SUBROUTINE INVERT(LR,LA,IG,IX2,IY2,NX,NY,IAN,N)
      IMPLICIT INTEGER*2 (I,J,K,L,M,N)
-THIS ROUTINE INVERTS THE TRANSFORM PERFORMED BY 'HOUGH'. IT
-EXTRACTS FROM THE RHO/THETA MATRIX THOSE CELLS EXCEEDING THE
-THE THRESHOLD 'IG'. FOR EACH EXTRACTED CELL IT CALCULATES
-THE CENTER POINT (X&Y), HALF-LENGTH, SLOPE & INTERCEPT, THEN
-STORES THESE IN THE ARRAY 'IL'. THE LINE IS THEN BACK-
-PROJECTED ONTO THE EDGE FIELD 'IMG' FOR OPTIONAL DISPLAY.
-THE COORDINATE SYSTEM USES THE UPPER-LEFT CORNER AS THE ORI-
-GIN, BUT IS ROTATED 90 DEGREES FOR LINES WITH ANGLES BETWEEN
-PI/4 AND 3PI/4.
      COMMON /HUFF/HF(13,256,4),NTOT(256),XX/CH/IMG(256,256)
      COMMON /LINE/IL(128,6),SLP(128),CPT(128)/LT/A(13,3),PI
      INTEGER*2 HF
      CHARACTER*1 IMG,IAN
      N=0
      IF(IAN.NE.'Y') GO TO 99
      WRITE(6,98) IX2,IY2,IG
   98 FORMAT('0','INVERTED IMAGE, COORDS',2I5,' THRESH.=',I4)
   99 DO 20 K=1,LA
      DO 20 J=1,LR
      IF(HF(K,J,1).LT.IG) GO TO 20
      N=N+1
      IL(N,1)=HF(K,J,1)
      IL(N,2)=(K-1)*180./LA
      IL(N,3)=HF(K,J,3)/IL(N,1)
      IL(N,5)=ABS(IL(N,3)-HF(K,J,2)/3)+1
      IL(N,6)=HF(K,J,4)
      CPT(N)=(J-A(K,3))/XX
```

```
      SLP(N)=-TAN(A(K,2))
      IL(N,4)=SLP(N)*IL(N,3)+CPT(N)
      IF(IAN.EQ.'N') GO TO 20
      DO 15 M=IL(N,3)-IL(N,5),IL(N,3)+IL(N,5)
      IY=SLP(N)*M+CPT(N)
      IF(M.LT.1.OR.IY.LT.1) GO TO 15
      IF(A(K,1).GT.(PI/4.).AND.A(K,1).LT.(.75*PI)) GO TO 13
      IF(M.LE.NY.AND.IY.LE.NX) IMG(IY,M)='X'
      GO TO 15
   13 IF(M.LE.NX.AND.IY.LE.NY) IMG(M,IY)='X'
   15 CONTINUE
   20 CONTINUE
      IF(IAN.EQ.'N') GO TO 30
      DO 18 K=1,NX
   18 WRITE(6,*) (IMG(K,J),J=1,NY)
   30 RETURN
      END
      SUBROUTINE PARLINES(N,IX,IY,ST,OT,DT,BT,FNAME,LH,IAN,MP,NW)
      IMPLICIT INTEGER*2 (I,J,K,L,M,N)
      COMMON /LINE/IL(128,6),SLP(128),CPT(128)/BC/IC(128,8),ID(128)
      CHARACTER*10 BRD
      CHARACTER*1 IAN
      CHARACTER*9 FNAME
-THIS ROUTINE EXTRACTS FROM THE SET OF LINES GENERATED BY
-'INVERT' THOSE SETS OF LINES THAT MAY BE BRIDGES. THESE ARE
-SETS OF LONG, PARALLEL LINES OF APPROX. EQUAL LENGTH AND
-LOCATION. IF IAN='Y', THE SURVIVING CANDIDATES ARE THEN
-TESTED BY 'GLOBCON' TO BE OVER 'WATER' AND THUS A BRIDGE.
      IF(MP.EQ.1) GO TO 99
      WRITE(6,1)
    1 FORMAT('0',' CTR(X,Y)  ANGLE LNGTH WIDTH #LINES RESULT')
   99 DO 2 K=1,N
    2 ID(K)=0
      NP=0
      BRD='          '
      DO 20 K=1,N-1
      IF(ID(K).EQ.1) GO TO 20
      IE=1
      DO 9 J=K+1,N
      IF(IL(K,2).NE.IL(J,2).OR.ID(J).NE.0) GO TO 9
      A=ATAN(SLP(K))
      D=(COS(A)*(CPT(K)-CPT(J)))**2
      AVG=IL(K,5)+IL(J,5)
      X=IL(K,3)-IL(J,3)
      Y=IL(K,4)-IL(J,4)
      OF=X**2 + Y**2 - D
      IF(D.GT.(ST**2).OR.OF.GT.((OT*AVG)**2)) GO TO 9
      IF(ABS((IL(K,5)-IL(J,5))*2.).GT.(DT*AVG)) GO TO 9
      ID(K)=2
      ID(J)=1
      IE=IE+1
      IF(IE.GT.2) GO TO 5
      NP=NP+1
      IF(IL(K,2).GT.45.AND.IL(K,2).LT.135) GO TO 3
```

```
      IC(NP,1)=IL(K,3)
      IC(NP,2)=IL(K,4)
      GO TO 4
    3 IC(NP,1)=IL(K,4)
      IC(NP,2)=IL(K,3)
    4 IC(NP,3)=IL(K,2)
      IC(NP,4)=IL(K,5)
      IC(NP,7)=IL(K,1)
    5 IF(IL(K,2).GT.45.AND.IL(K,2).LT.135) GO TO 6
      IC(NP,1)=IC(NP,1)+IL(J,3)
      IC(NP,2)=IC(NP,2)+IL(J,4)
      GO TO 7
    6 IC(NP,1)=IC(NP,1)+IL(J,4)
      IC(NP,2)=IC(NP,2)+IL(J,3)
    7 IC(NP,4)=IC(NP,4)+IL(J,5)
      IC(NP,7)=IC(NP,7)+IL(J,1)
      D=ABS(D)
    8 IC(NP,5)=SQRT(D)
    9 CONTINUE
      IF(ID(K).EQ.0) GO TO 20
      IC(NP,1)=IC(NP,1)/IE +IY-1
      IC(NP,2)=IC(NP,2)/IE +IX-1
      IC(NP,4)=IC(NP,4)/IE * 2
      IC(NP,6)=IE
      IC(NP,7)=IC(NP,7)/IE
      DO 13 J=1,NP-1
      IF((ABS(IC(NP,1)-IC(J,1))+ABS(IC(NP,2)-IC(J,2))).GT.5)
     * GO TO 13
      L=0
      IF(IC(NP,6).GT.IC(J,6)) L=L+1
      IF(IC(NP,7).GT.IC(J,7)) L=L+1
      IF(IC(NP,4).GT.IC(J,4)) L=L+1
      IF(L.GE.2) GO TO 12
      NP=NP-1
      GO TO 20
   12 IC(J,6)=0
   13 CONTINUE
   20 CONTINUE
      DO 25 K=1,NP
      IF(IC(K,6).EQ.0) GO TO 25
      IF(IAN.EQ.'Y') CALL GLOBCON(BT,LH,FNAME,NW,K,BRD)
      WRITE(6,24) (IC(K,L),L=1,6),BRD
   24 FORMAT(' ',I4,',',I4,4I6,2X,A10)
   25 CONTINUE
      RETURN
      END
      SUBROUTINE GLOBCON(BT,LH,FNAME,NW,NP,BRD)
      IMPLICIT INTEGER*2 (I,J,K,L,M,N)
-THIS ROUTINE PERFORMS A GLOBAL CONTRAST TEST ABOUT A
-BRIDGE 'CANDIDATE'. IT COMPARES THE UNIFORMITY WITHIN
-A NW X NW WINDOW AT PTS ON EITHER SIDE OF THE 'BRIDGE'. IF
-THE UNIFORMITY OF BOTH SIDES ARE BELOW A THRESHOLD(BT)
-AND THE MEAN BRIGHTNESS ON BOTH SIDES ARE APPROX. EQUAL,
-THE CANDIDATE IS JUDGED TO BE OVER WATER, THUS A 'BRIDGE'
```

```
      COMMON /BC/IC(128,8),ID(128)
      CHARACTER*10 BRD
      CHARACTER*9 FNAME
      BRD='BRIDGE    '
      PI=ATAN(1.0)*4.
      ANG=IC(NP,3)*PI/180.
      DIST=(NW*2+2+IC(NP,5)/2.)
      IXL=IC(NP,1)-DIST*SIN(ANG)
      IYL=IC(NP,2)-DIST*COS(ANG)
      IXR=IC(NP,1)+DIST*SIN(ANG)
      IYR=IC(NP,2)+DIST*COS(ANG)
      CALL TEX(LH,IR,FNAME,IXL,IYL,E1,V1,NW,BRD)
      CALL TEX(LH,IR,FNAME,IXR,IYR,E2,V2,NW,BRD)
      IF(BRD.EQ.'INCONCLUS.') GO TO 20
      DM=ABS((E1-E2)/2.)
      IF(V1.GT.BT.OR.V2.GT.BT.OR.DM.GT.BT) BRD='NOT BRIDGE'
   20 RETURN
      END
      SUBROUTINE TEX(LH,IR,FNAME,I,K,E,V,NW,BRD)
      IMPLICIT INTEGER*2 (I,J,K,L,M,N)
      CHARACTER*10 BRD
      CHARACTER*9 FNAME
      DIMENSION IR(LH)
-THIS ROUTINE CALCULATES THE MEAN BRIGHTNESS AND UNIFORMITY
-(IE, STD.DEV.OR.BRIGHTNESS) WITHIN A NWXNW WINDOW CENTERED
-ON I,K. THIS 'TEXTURE' MEASURE IS USED IN THE GLOBCON TEST.
      E=0
      V=0
      OPEN(ACCESS='DIRECT',NAME=FNAME,TYPE='OLD',UNIT=21)
-THE NEXT LINE SEES IF THE POINT IS TOO CLOSE TO THE EDGE.
      IF(K.LE.NW.OR.I.LE.NW.OR.I.GT.(LH-NW)) GO TO 10
      WS=(2*NW+1)**2
      DO 5 L=K-NW,K+NW
      READ(21'L,END=10) IR
      DO 5 M=I-NW,I+NW
      E=E+IR(M)
    5 V=V+FLOAT(IR(M))**2
      E=E/WS
      V=SQRT(V/WS -E**2)
      GO TO 20
   10 BRD='INCONCLUS.'
   20 CLOSE(UNIT=21)
      RETURN
      END
$
```



To cope with the expanding technology, our society must be assured of a continuing supply of rigorously trained and educated engineers. The School of Engineering and Applied Science is completely committed to this objective.